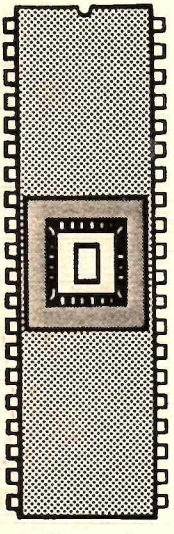
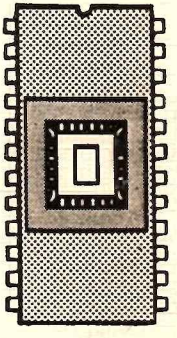
LOGICAL DEVICES, INC.





SHOOTER™

EE/EPROM PROGRAMMER

INSTRUCTION MANUAL

LOGICAL

TABLE OF CONTENTS

1.Ø UNPACKING

1) Carefully unpack unit from the shipping container and inspect for possible shipping damage.

2) Check to see if the following items have been included:

a) SHOOTER EPROM PROGRAMMER
b) Set of 3 configurators for 2716, 2732, 2732A
c) Manual
d) Warranty registration
e) Serial communication cable (without RS-232 connector)

3) Plug the unit into a 115 vac outlet. The front panel status LED'S will light in sequence (red ERROR, yellow BUSY, then green READY).
The green LED will stay on to indicate that the unit is ready to accept a command.

2.Ø INTRODUCTION

The SHOOTER programmer is a stand-alone unit complete with cabinet and power supply. There are three basic modes of operations. Stand-alone, Terminal controlled, and Computer controlled.

2.1 STAND ALONE MODE

In the stand alone mode of operation you can copy and verify EPROMs by the use of the front panel function switches. There is no need for a computer or terminal hook-up with this mode.

1) Plug the unit into an AC outlet

2) Insert the proper Configurator plug in the CNF socket with the white dot to the left. Refer to section 5.1 to choose the proper configurator plug.

3) Insert the master EPROM (the one you wish to copy) into the ZIF socket. Align the EPROM so that pin one is in the upper left same side as the ZIF handle and the ground pin (pin 12 for 24 pin EPROMS, and pin 14 for 28 pin EPROMS) is in pin 14 (lower left corner) of the ZIF socket. Refer to Appendix E on how to insert EPROMs.

4) Press the front panel RESET switch. This will reset the system and automatically read the entire contents of the master EPROM into the SHOOTER'S RAM buffer. You will see the LEDs go on in sequence as in the power-up procedure.

5) Remove the master EPROM and replace it with an erased EPROM of the same type.

NOTE

In the stand alone mode, to be positively sure that an EPROM is properly erased:

a. Insert a known blank EPROM inserted in the unit and press the RESET switch;
b. Insert the EPROM in question into the unit:
c. Press the VERIFY switch. If the red L.E.D. goes on, the EPROM is not completely erased.

6) Check to ensure that the EPROM and the Configurator plug are inserted in their respective sockets with their proper orientation. This is a very important step to prevent any damage to the device.

C A U T I O N

BEFORE CONTINUING TO PROGRAM, BE SURE THAT THE PREVIOUS STEPS WERE PERFORMED CORRECTLY OR YOU MAY DAMAGE YOUR DEVICE.

7) Depress the PROG switch on the front panel. The BUSY (Yellow L.E.D.) indicator will remain on during programming. When the program cycle is complete, the READY (Green L.E.D.) will be activated. If the EPROM does not program properly, the ERROR (Red L.E.D.) indicator will also go on.

8) To insure that every location has been properly programmed, the Verify feature can be utilized by pressing the VERIFY switch on the front panel. This will initiate a comparison of every data byte in the EPROM against the SHOOTER'S RAM. If there is any discrepancy, the red ERROR L.E.D. will go on.

2.2 TERMINAL MODE

The SHOOTER has the ability to operate directly with a terminal. An internal communications program allows you to utilize a terminal to control the SHOOTER.

1) Connect the unit to a serial terminal or a computer through the serial communication connector in the back of the unit. See appendix C)

2) Set the dip switches on the rear panel the the desired baud rate. See section 6.Ø for switch settings.

3) The SHOOTER sends a command menu to the terminal followed by a prompt "*" upon power-up or reset. A command may be entered by the user after the prompt appears. See section 3.Ø for command descriptions.

C = CHECKSUM
F = FILL RAM
G = HEX DUMP
H = HEX LOAD
K = BINARY DUMP
L = BINARY LOAD
N = BLANK CHECK
P = PROGRAM
R = READ EPROM
T = DOWNLOAD
U = UPLOAD
V = VERIFY
W = UPLOAD WITH WAIT
X = EXAMINE

? = MENU
I = INTEL HEX
M = MOTOROLA HEX
O = OFFSET

NOTE: An "*" appears beside either Intel Hex or Motorola Hex to indicate which mode the SHOOTER is in.

4) You may display the menu at any time by typing a question mark.

5) After the completion of any command the SHOOTER will transmit the character sequence "^F*EOJ*" or "^U*EOJ*". The character "^" preceding any character indicates the control code for that character. A "^F" (HEX Ø6) indicates an ACK (short form for Acknowledge) meaning that all went well in the previous instruction. A "^U" (HEX 15) transmitted indicates a NAK (short form for Negative Acknowledge) meaning that an error was encountered on the previous instruction. The codes $Ø6 or $15 are typically non-displaying characters that would only be used when connected to a host computer.

NOTE: "^U*EOJ*" transmission sequence means that the control U followed by ASCII characters *EOJ* is sent.

2.4 COMPUTER MODE

This mode of operation is similar to the terminal mode, with the difference that you can now transfer HEX files to and from the SHOOTER. The maximum RAM buffer size on the SHOOTER is 128K bits, or 16K bytes. The computer mode consists of two modes of operations:

1) HOST TERMINAL MODE - In this mode of operation the computer is made to look like a terminal to the SHOOTER. The user can operate the SHOOTER as described in section 2.2. To make your computer look like a terminal you must instruct your computer via a software program. This type of software program is generally referred to as a "TERMINAL" or "MODEM" program and is generally available commercially under various names; "CROSS-TALK" and "MODEM-7" are two examples.

NOTE: The serial cabling to a computer is generally different than to a CRT terminal. Refer to section 6.Ø for wiring details.

2) FILE TRANSFER MODE - This mode of operation allows you to transfer a data file to the SHOOTER or read a data file from the SHOOTER. Refer to apendix A for details on hex formats.

3.Ø SHOOTER COMMAND DICTIONARY

C CHECKSUM	: H HEX LOAD
F FILL RAM	: G HEX DUMP
N BLANK CHECK	: L BINARY LOAD
P PROGRAM	: K BINARY DUMP
R READ EPROM	: T DOWNLOAD
V VERIFY	: U UPLOAD
X EXAMINE MEMORY	: W UPLOAD WITH WAIT
? MENU	: O OFFSET ØØ
I INTEL HEX *	
M MOTOROLA HEX	

3.1 C = CHECKSUM

The CHECKSUM command displays a 16 bit sum of the RAM contents. The RAM size is specified by the configurator plug.

3.2 F = FILL RAM

This command allows you to fill the entire RAM (as specified by the configurator plug) with any desired pattern. To initiate the FILL function, enter the character "F" through the RS232 port, followed by two hex digits representing the desired pattern.

3.3 G = HEXADECIMAL UP-LOAD

When directed, the SHOOTER will transmit the contents of it's internal RAM to the terminal or computer. The amount that will be transmitted will be automatically determined by the configurator plug in the CNF socket.

An example of this would be: 2716=2K

1) To command the SHOOTER to enter the Hex up-load mode, type a "G".

2) The SHOOTER will now transmit the contents of it's internal RAM buffer in hexadecimal format.

3.4 H = HEXADECIMAL DOWN-LOAD

1) To command the unit to enter the Hex DOWN-LOAD mode, type a "H".

2) If no character is received within approximately 7 seconds, the unit will exit the DOWN-LOAD mode and return to the READY state.

3) Data are stored in the SHOOTER's Ram buffer starting at location ØØ.

3.5 K = BINARY UP-LOAD

When directed, the SHOOTER will transmit the contents of it's internal RAM to the terminal or computer. The amount that will be transmitted will be automatically determined by the configurator plug in the CNF socket.

An example of this would be: 2716=2K

1) To command the SHOOTER to enter the up-load mode, type a "K".

2) The SHOOTER will now transmit the contents of it's internal RAM buffer in Binary format.

3.6 L = BINARY DOWN-LOAD

1) To command the unit to enter the DOWN-LOAD mode, type a "L".

2) If no character is received within approximately 7 seconds, the unit will exit the DOWN-LOAD mode and return to the READY state.

3) Data will be loaded into the SHOOTER's Ram buffer beginning at the location specified by the offset register.

3.7 N = BLANK CHECK

The BLANK CHECK command allows you to check an unknown EPROM for the blank data pattern before issuing the PROGRAM command. To initiate this command enter the character "N" on the terminal keyboard. If the EPROM is not erased the SHOOTER will respond with the character sequence "NB ERROR" and light the ERROR L.E.D.

3.8 P = PROGRAM

When the unit is directed to program the EPROM, a simple transfer takes place; the contents of RAM are programmed into the coinciding locations of the EPROM. The EPROM must have been previously erased by the "QUV-T8 EPROM ERASER" or equivalent.

1) Type a "P" on the terminal. The BUSY LED will light.

2) After the EPROM has been completely programmed, all locations are then compared to the RAM. If any errors are found, (see VERIFY) the ERROR L.E.D. will light. If you are using the INTELIGENT programming algorithm the program cycle will stop at the first unprogrammable location.

3) If an error occurs, check the configurator plug for proper type and proper insertion in the CNF socket.

3.9 R = READ EPROM

The READ command allows you to read the EPROM into SHOOTER'S internal RAM. Make sure that the EPROM is inserted correctly and the proper configurator plug is inserted in the CNF socket before issuing the READ command.

3.1Ø T = FORMATTED HEXADECIMAL DOWN-LOAD

1) To command the unit to enter the DOWN-LOAD mode, type a "T".

2) The unit is now expecting the header character of the specified format, an "S" for MOTOROLA, or an ":" for INTEL. At this point all other characters will be ignored until the header is received.

3) If no character is received within approximately 7 seconds, the unit will exit the DOWN-LOAD mode and return to the READY state.

4) The most signifigant bit of the address will be ignored, and any references to address space from 4ØØØH to 7FFFH will not write to RAM. When the address is received the contents of the offset register (see OFFSET command) will be subtracted from it and the resulting address will be the actual RAM address where the data will be stored.

5) The checksum is compared at the end of each data record. If an error is detected the error L.E.D. will light but the data will still be stored in RAM and the unit will still remain in this mode until the end of the record is found or until transmission stops.

3.11 U = FORMATTED HEXADECIMAL UP-LOAD
When directed, the SHOOTER will transmit the contents of it's internal RAM to the terminal or computer. The amount that will be transmitted will be automatically determined by the configurator plug in the CNF socket.

An example of this would be: 2716=2K

1) To command the SHOOTER to enter the up-load mode, type a "U".

2) The SHOOTER is now expecting two more characters. These characters will be two hexadecimal digits which are used to offset the SHOOTER'S internal RAM address to conform to the host computer's memory map. Later, when the uploaded hex file is used on the host computer, it will be located at a valid memory location and not 'Ø', which on many host computers is not usable as a data area.

EXAMPLE:
The following is an example of a 4K byte Up-Load

from a 2732.

UPLOAD WITH OFFSET COMMAND	STARTING ADDRESS IN THE COMPUTER	FINAL ADDRESS
UØØ	ØØØØ	ØFFF
U3Ø	3ØØØ	3FFF
U96	96ØØ	A5FF
UCF	CFØØ	DEFF

3) The SHOOTER will now transmit the contents of it's internal RAM buffer in the hexadecimal format previously specified. If you do not change the format after power up, INTEL format will be used. A 7 second delay before transmission is incorporated in order to allow setup time for the host computer.

3.12 V = VERIFY

This feature, whether directed from the terminal or the front panel switch, initiates a location by location comparison of the RAM buffer against the EPROM.
If a discrepancy occurs, the RED LED will go on. If all data compares, the GREEN LED goes on.
This is a very useful feature for comparing a large number of EPROMs against a known master.

3.13 W = UPLOAD WITH WAIT

This function is useful mainly when using the CPM PIP command to UPLOAD to a host computer. The WAIT option causes a 25 second delay before transmission to allow the PIP command (or equivalent). In addition to a delay the transfer is terminated with a control Z (1A hex) which is necesary to close the file correctly.

3.14 X = EXAMINE
The EXAM command allows you to gain direct access to a specific memory location in SHOOTER'S internal RAM, and if desired, change the contents of that location.

1) To use this command type "X".

2) A space will then be displayed.

3) The SHOOTER is now waiting for a 4 digit hex address. The specified address must be within the range of the EPROM being programmed.

4) After the address has been entered, the SHOOTER will display the address, an "=" sign, and the contents of this location.

5) If no change is to be made to this location a "SPACE" will increment the address and display the contents of the next location. To change the data, simply enter new data (two Hex digits).

6) To terminate the EXAM command, type a "CR".

Example: Examine location Ø2ØØ and change location Ø2Ø1 from 67 to 55
verify that location 2Ø1 contains 55.

User entry is in bold type.

```
* X Ø2ØØ
Ø2ØØ=45 <SPACEBAR>
Ø2Ø1=67 55
Ø2Ø2=7A <CR>
*
* X Ø2Ø1
Ø2Ø1=55 <CR>
*
```

3.15 ? = MENU

You may recall the SHOOTER command menu at any time by entering a question mark from the terminal.

3.16 I = INTEL HEX
An "I" entered via the RS-232C port will direct all file transfer communications to be carried out in the INTEL hexadecimal format.
See Appendix A for details and examples.

Note: The SHOOTER will default to INTEL upon power up.

3.17 M = MOTOROLA HEX
An "M" entered via the RS-232C port will direct all communications to be carried out in the MOTOROLA hexadecimal format. See Appendix A.

3.18 O = OFFSET

The OFFSET function is used in conjunction with the DOWNLOAD command and once set it will retain its value until another OFFSET command is issued. When downloading formatted hex records the OFFSET value is subtracted from the most significant byte of each record;

```
EXAMPLE: Record start address.....Ø5ØØ
         OFFSET value.............Ø5
         RAM address.............ØØØØ
```

NOTE:
The OFFSET value defaults to ØØ on power up.

4.0 STATUS LED DISPLAYS

THERE ARE THREE LED INDICATORS ON THE FRONT PANEL OF THE UNIT USED EITHER FOR STAND-ALONE MODE OR RS-232 MODE. THE FOLLOWING DESCRIBES THE FUNCTIONS OF THE DISPLAYS:

4.1 ERROR L.E.D. (RED)

THIS L.E.D. WILL LIGHT WHEN THE FOLLOWING FAULTS OCCUR:

A. AN EPROM DID NOT PROGRAM CORRECTLY. TRY ERASING THE EPROM AGAIN FOR A LONGER PERIOD OF TIME. ALSO, CHECK THE CONFIGURATOR PLUG.

B. AN ERROR WAS FOUND WHEN PERFORMING A DOWN-LOAD. CHECK BAUD-RATE SELECTION. CHECK FOR NON-VALID DATA IN THE HOST COMPUTER'S HEX FILE.

4.2 BUSY L.E.D. (YELLOW)

THIS L.E.D. WILL LIGHT WHEN EVER THE SHOOTER'S CPU IS ENGAGED IN A COMMAND.

W A R N I N G

DO NOT ATTEMPT TO INSTALL OR REMOVE AN EPROM ANY TIME THAT THIS L.E.D. IS ON. OR THE DEVICE MAY BE DESTROYED.

4.3 READY L.E.D. (GREEN)

THIS L.E.D. WILL LIGHT WHEN THE UNIT IS IN IDLE STATE. ALL POWER, ADDRESS, AND DATA LINES HAVE BEEN TAKEN LOW (I.E. LESS THAN 1 VOLT) MAKING IT SAFE TO REMOVE OR INSTALL THE EPROM AT THIS TIME.

4.4 ERROR CODES

SOME COMMANDS WILL SEND AN ERROR MESSAGE TO THE TERMINAL INDICATING A FAILED CONDITION. THE FOLLOWING IS A LIST OF THESE FUNCTIONS AND THE CORESPONDING ERROR MESSAGES.

BLANK CHECK.....NB ERROR
PROGRAM (INTELIGENT ONLY).....PF ERROR
VERIFY.....NV ERROR

AFTER EACH COMMAND IS COMPLETED, THE SHOOTER WILL SEND EITHER THE CHARACTER SEQUENCE "*EOJ*" PRECEDED BY A HEX 06 (THE CODE FOR ACK) IF THE FUNCTION PASSED, AND A HEX 15 (CODE FOR NAK) IF IT FAILED.

5.0 DEVICE CONFIGURATION

IN ORDER TO CONFIGURE THE SHOOTER FOR A PARTICULAR DEVICE TYPE YOU MUST PLUG ONE OF THE SMALL DIP CONFIGURATOR PLUGS SUPPLIED WITH THE SHOOTER INTO THE FRONT PANEL CNF SOCKET. THESE PLUGS ARE CODED WITH THE GENERIC NUMBER REPRESENTING THE DEVICE TYPE.

YOU CAN CHANGE THE CONFIGURATOR PLUG WHILE POWER IS ON WITHOUT DAMAGE TO THE UNIT BUT NOT WHILE THE BUSY (YELLOW) LED IS ON. IT IS POSSIBLE TO EDIT THE RAM BUFFER WITHOUT THE CONFIGURATOR PLUG INSERTED.

YOU CAN TRANSFER DATA FROM ONE TYPE OF EPROM TO ANOTHER TYPE BY CHANGING CONFIGURATOR PLUGS AFTER THE READ OPERATION IS COMPLETE. IF YOU LOSE YOUR CONFIGURATOR YOU CAN ALWAYS OBTAIN MORE FROM LOGICAL DEVICES, INC. DO NOT INSERT ANY OTHER TYPE OF HARDWARE IN THE CNF SOCKET AS IT MAY DAMAGE THE SOCKET.

I M P O R T A N T

FOR SOME EPROMS YOU WILL HAVE TO SET THE DIP SWITCH ON THE BACK OF THE UNIT (SWITCH #3) IN ADDITION TO THE CONFIGURATOR. THIS SWITCH CONTROLS THE VCC VOLTAGE (5V OR 6V) AND THE TYPE OF PROGRAMMING ALGORITHM (STANDARD OR INTELIGENT) USED. IN THE DEVICE DICTIONARY THE POSITION OF THIS SWITCH IS INDICATED IN THE COLUMN MARKED DIP SWITCH 3 AND IS DOWN FOR INTELIGENT AND UP FOR STANDARD PROGRAMMING. IF YOU ARE USING THE 27256 EPROM SWITCH #4 WILL BE USED AS THE UPPER ADDRESS BIT (A14). THE REASON FOR THIS IS THAT THE 27256 EPROM IS A 256K BIT DEVICE AND THE SHOOTER HAS ONLY 128K BITS OF RAM BUFFER, THEREFORE PROGRAMMING MUST BE DONE IN TWO PASSES. THE POSITION OF SWITCH 4 IS UP FOR THE UPPER HALF AND DOWN FOR THE LOWER HALF.

TO USE THE DEVICE TABLE SIMPLY LOCATE THE TYPE OF DEVICE OR THE EQUIVALENT ON THE LEFT COLUMN. THEN DETERMINE THE CONFIGURATOR PLUG NUMBER YOU MUST INSERT IN THE CNF SOCKET UNDER THE CNF COLUMN. INSERT YOUR EPROM IN THE ZIF SOCKET.

5.1 DEVICE DICTIONARY

The SHOOTER can support the following devices:

DEVICE SIZE	DEVICE TYPE	CNF #	STANDARD SUPPORT	SWITCH 3#	SWITCH 4#
Advanced Micro Devices (AMD):					
2K X 8 :	AM2716DC,AM2716nDC (all)	#2716	YES	UP	UP
4K x 8 :	AM2732	#2732	YES	UP	UP
8K x 8 :	AM2764	#2764	M128	DOWN	UP
Fairchild:					
4K X 8 :	F2732	#2732	YES	UP	UP
Fujitsu:					
4K X 8 :	MBM2732	#2732	YES	UP	UP
8K X 8 :	MBM2764	#2764	M128	DOWN	UP
8K X 8 :	MBM27C64	#2764	M128	DOWN	UP
Hitachi:					
2K X 8 :	HN462716G	#2716	YES	UP	UP
4K X 8 :	HN462732G	#2732	YES	UP	UP
4K X 8 :	HN462732AG	#2732A	YES	UP	UP
8K X 8 :	HN482764G	#2764	M128	DOWN	UP
Intel Corp.:					
2K X 8 :	2716	#2716	YES	UP	UP
4K X 8 :	2732	#2732	YES	UP	UP
4K X 8 :	2732A	#2732A	YES	UP	UP
4K X 8 :	8751	#8751	M128	UP	UP
8K X 8 :	2764	#2764	M128	DOWN	UP
16K X 8 :	27128	#27128	M128	DOWN	UP
16K X 8 :	27128A	#128A	M128	DOWN	UP
32K X 8 :	27256 (HIGH BYTES)	#27256	M128	DOWN	UP
	(LOW BYTES)				DOWN
Mitsubishi:					
2K X 8 :	M5L2716K	#2716	YES	UP	UP
4K X 8 :	M5L2732K	#2732	YES	UP	UP
Mostek:					
2K X 8 :	MK2716-n (all)	#2716	YES	UP	UP

DEVICE SIZE	DEVICE TYPE	CNF #	STANDARD SUPPORT	SWITCH 3	SWITCH 4
MOTOROLA:					
2K X 8 :	MCM2716, MCM27L16	#2716	YES	UP	UP
4K X 8 :	MCM2532	#2532	M128	UP	UP
NATIONAL SEMICONDUCTOR:					
2K X 8 :	MM2716E, MM2716M, NMC27C16	#2716	YES	UP	UP
4K X 8 :	NMC2732	#2732	YES	UP	UP
8K X 8 :	MM27C64	#2764	M128	DOWN	UP
TEXAS INSTRUMENTS:					
2K X 8 :	TMS 2516	#2716	YES	UP	UP
4K X 8 :	TMS 2532	#2532	M128	UP	UP

*NOTE: SWITCH 4 CONTROLS THE MSB ADDRESS LINE FOR 27256 EPROMS WHEN THE 27256 CONFIGURATOR IS IN; OTHERWISE DOWN POSITION DOES NOT SKIP FF's (USED TO ERASE EEPROMS). SWITCH 3 DOWN SELECTS THE INTELLIGENT PROGRAMMING ALGORITHM. THIS CHART IS AS ACCURATE AS POSSIBLE WITH DATA AVAILABLE. CHECK THE DATA SHEET FOR YOUR PROMS TO SEE IF INTELLEGENT PROGRAMMING IS APPROVED.

#2816	AR6	MODULE	-	DOWN
#2864			UP	DOWN

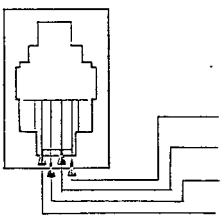
6.Ø SERIAL INTERFACE

SHOOTER has a built-in serial I/O. The serial communication connector is located on the rear of the SHOOTER. See Appendix C. for more information about the connector wiring.

The serial I/O can be directly connected to a CRT terminal (with pin 5 left disconnected). For most computer interfaces RS232 pins 2 and 3 must be reversed. It is always a good idea to refer to the pin-out of the serial interface for each terminal or computer to make sure that the wiring is correct.

RS-232C PIN ASSIGNMENT

SHOOTER RS-232 PIN#



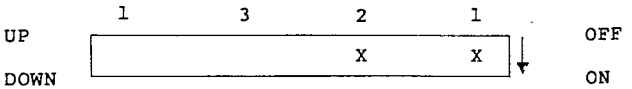
4 = Rx data (input to shooter)
3 = Signal Ground
2 = CTS (input to shooter)
1 = Tx data

NOTE: SHOOTER provides an, internal pull-up resistor for the CTS line providing a user option to exercise this handshake line. Certain development systems may require long delays in responding to the serial port, in which case the CTS line to SHOOTER must be used. If CTS line is not activated by the computer or terminal, disconnect pin 5 from the RS232 connector cable to avoid accidently exercising this handshake line. Some computers require RS-232 pins 4 and 5 and RS-232 pins 6, 8 and 2Ø to be connected together to properly condition the handshaking lines that are not used.

SETTING THE BAUD RATE

Prior to establishing communications between a terminal or computer and the SHOOTER, the baud rate switch on the rear panel should be set to the same baud rate as the terminal or computer;

The baud rate switches are switches 1 and 2 of the four position DIP switch on the rear of the unit.



BAUD RATE	SW 1	SW 2
110	DN/ON	DN/ON
300	UP/OFF	DN/ON
1200	DN/ON	UP/OFF
2400	UP/OFF	UP/OFF

SERIAL DATA FORMAT:

Half-Duplex
One start bit
8-data bits
No Parity
Two stop bits

SERIAL SIGNAL LEVELS; +12V,-12V (CTS ENABLED = + 12V, NOT CTS = -12V)

7.Ø SOFTWARE INTERFACE

In order to establish communication between the SHOOTER and a computer, a computer RS-232C serial port (11Ø, 3ØØ, 12ØØ, or 24ØØ baud) must be provided.
In a typical development cycle the user assembles the program into an object file and then transfers the object file from the memory or disk to the SHOOTER.
Some software development tools such as assemblers and compilers already generate the output of the assembly in the form of INTEL or MOTOROLA hex formats. In such a case the job of sending a file to the PROM PROGRAMMER is only a matter of instructing the computer to transfer the file from the disk (source) to the Serial port (destination).

Most operating systems provide a utility command to facilitate file transfer operations. For example, under the CPM operating system, the "PIP" command is used to do such a task. Under PC DOS you may use the COPY command. If your development system does not generate the standard formats, then you must provide a program to convert a binary or straight ASCII file to the standard HEX formats.
The following software drivers are available from Logical Devices Inc. on discs:

IBM PC software driver
APPLE -II software driver
CPM 2.2 8" SS SD
Flex Ø9 5" SS SD
Commodore 64

Software drivers consist of two distinct parts;

1- A simple communications (terminal) mode.
2- Hex converter mode.

The above mentioned software drivers have one or both parts incorporated into them. For the most part, you may not even need a software driver should your system have a Terminal mode or Terminal program. For example, the APPLEII software drivers do not have terminal mode because the APPLEII computer can already act as a terminal with a few simple control characters.

In Appendix A you will find the listing of several types of microprocessors for generating the Intel or Motorola Hex formats. In some instances the listing provided (68Ø9 routines) is part of a larger program. It is provided to give you only an illustration of how it is done. There is also an example of a software driver for the TRS-8Ø color computer. You can use these routines as a guide to develop your own for any other computer system (see Appendix B).

Before you attempt to analyze the program code it is a good idea to know what the general structure of the software program should look like.

The following list defines the software structure for the drivers (a slightly modified version of a standard TERMINAL program).

1. Initialize Serial port.
2. Scan Serial port for input.
3. If input, display on terminal.
4. Scan terminal for user entry.
5. If no entry, go to step 2.
6. Output user entry to serial port.
7. If user entry is "T" go to download routine.
8. If user entry is "U" go to upload routine, or go to step 2.

DOWNLOAD ROUTINE

1. Prompt the user with the file name or memory addresses where the data is located.
2. Send either the "I" or "M" command to set the programmer in the proper hex mode.
3. Send an ASCII "T" (54H) to the serial port to indicate to the programmer that the next entry is going to be a valid hex file.
4. Send Hex file to the serial port.

UPLOAD ROUTINE

1. Prompt the user for a disk file name or a memory address where the uploaded data is to be stored.
2. Send to the serial port an ASCII "U" followed by two hex digits indicating the page offset for the load address in system RAM.
3. Go to the Hex load routine.
4. Return.

NOTE: Your driver routine address space must not coincide with the same address space as your upload file.

It is very important to insure that your serial interface is working properly. If you are using it for the first time there may be a chance that you may have a problem with it. READ the instruction manual for your serial port very carefully. Monitor the transmit and receive line with an osciloscope to insure that data is going out or coming in. In debugging any system always islolate the unknowns and deal with one unknown at a time.

It is also necessary to understand how the serial communication in your particular computer is accomplished.

Each system utilizes a slightly different hardware for the Serial/Modem port. For many systems a simple "Terminal/Modem" program is all that is required to operate the SHOOTER in the conversational mode. These programs are readily available for most systems. A terminal program is basically a program that makes your computer look like a CRT Terminal. All console entries are directed to the serial port and all data received from your serial port is displayed on the screen. Prior to sending and receiving data to and from the serial port you must set the following parameters:

Start Bits	1
Stop Bits	2
No. Data Bits	8
Parity	Ignored
Baud	11Ø, 3ØØ, 12ØØ, 24ØØ
Mode	Half Duplex

The above format is software selectable in certain systems. For example, in a Radio-Shack TRS-8Ø Model II with CPM 2.2 DOS, you can utilize the "SETUP"command to set your parameters. If you do not have a terminal program and wish to write your own, there are three basic ways to write such programs.

1. Terminal program written in BASIC language.
2. Terminal program using the callable routines in Bios or System Monitor.
3. Assembly language program with your own Serial Port Drivers.

All communication is done in ASCII Format. For example, if you were to direct the programmer to program an EPROM, you would store the ASCII equivalent of "P" (5ØH) in the data register of your serial controller.

The I/O addresses in many systems are memory mapped. That is: to transfer a byte to and from the I/O port would be similar to transferring a byte to and from a memory location.

If your system is not I/O memory mapped, you must use the Input/Output instructions of your microprocessor with the corresponding I/O# assignment. Most serial controller chips have two internal registers, one Data and one Control/Status register. If you are writing directly to the serial controller you must always test the condition of the ready status bit to insure that you do not over-run previous data.

7.1 USING CPM

Since CPM is the most common operating system for the microcomputers some examples of its use with the SHOOTER is provided here;
We assume that you have written or purchased a terminal program for your CPM system by the name of "MODEM.COM" Once you power up your system you should get the CPM prompt:

```
A>
                    [NOW TYPE "MODEM" AND RETURN]
A>MODEM <cr>
```

Once the program is activated it will prompt you for several options. Select the "terminal" option. If your SHOOTER is connected properly to the serial port, as soon as you depress the RESET switch on the front panel of SHOOTER, the menu should appear on your screen;

To set the programmer in the Intel hexadecimal mode, type "I".

```
*I
```

SHOOTER will respond with:

```
*EOJ*
*_
```

To examine and change a location in the SHOOTER RAM buffer, type "X" after the "*" prompt, followed by a four digit hex address:

```
*X ØØØ1
```

SHOOTER will respond with;

ØØØ1=hh (hh is a two digit hex value representing the contents of location ØØØ1).

```
Type new data or, space to skip;
ØØØ1=hh [SPACEBAR]
ØØØ2=hh [55] (as an example)
ØØØ3=hh [RETURN]  Return to prompt
*
```

Now if you go back and examine location ØØØ2 you should have the data value "55".

Once you have completed your editing of the RAM buffer you can exit the terminal mode program and return to the CPM operating system.

USING CPM PIP COMMAND TO DOWNLOAD

Once the format has been selected (Intel or Motorola),there are two ways to download a file to the SHOOTER; The easiest method is to append the SHOOTER ASCII commands to the beginning of your Hex file using the CPM editor:

Insert the string:

T

into the beginning of the file.

If you use the type command to display the file;

```
A>TYPE EZ.HEX        (EZ.HEX is the name of the file
                          as an example)
```

You should see;

T

```
:100000005C541D.....    (and the rest of your Intel
                           hex file)
```

Now use the PIP command to transfer the file to the serial port;

```
A>PIP PUN=A: EZ.HEX <CR>
```

You can change the device assignment to whatever you wish using the STAT VAL or SETUP command of the CPM.

The second way to download is to use your editor to create a permanent file containing only one character; the "T". Then using the SUBMIT facility of CPM, send the file containing the "T", and your Hex file with only one command (it is a good idea to insert nulls between files). The advantage of this method is that you will not have to modify your Hex file after every assembly.

EXAMPLE:

```
 PIP PUN: = T.DWN,NUL:,EZ.HEX
```

If you want to modify the format mode in the Shooter, create another permanent file containing either an "I" or "M" then using the SUBMIT facility, send this file followed by the two described above.

NOTE** Some 8080 Assemblers do no generate the proper Intel Hex Format End of File character (Ten 00 characters are generated instead). The SHOOTER will generate an error if this last line of zeros is encountered. There are several ways to get around this problem, the easiest way is to delete the last line of the Hex File (Ten 00's) using the Editor. SHOOTER will time out (7 seconds) after file transfer is completed. Also, you may try using the I parameter of the PIP command to ignore the ":00" records of Hex Format.

CPM USERS

Many software development tools such as assemblers or cross assemblers have an option that allows you to specify the Hex address differently than the actual assembly (ORIGIN) address. SHOOTER loads the Intel Hex file in its RAM buffer at the specified address of Hex format. Since the programmer always programs the EPROM from location ØØØØ, you must make sure that the starting address of your file is directed to location ØØØØ of SHOOTER'S RAM buffer. You may do this by creating another file which contains the character "O" (OFFSET COMMAND) followed by two hex digits equal to the PAGE number which your hex file will start. Then simply send this file to the programmer before the hex file transfer.

USING CPM PIP COMMAND TO UPLOAD

You can also use the CPM PIP command to UPLOAD a file to a host computer. However instead of using the character "U" to initiate the UPLOAD function, the character "W" should be used (see UPLOAD with WAIT command). This will cause a 25 second delay before transmission to allow time for the CPM PIP command to execute. It will also terminate the transmission with a control Z, which is necessary for CPM to close the file correctly.

8.Ø TROUBLESHOOTING

SHOOTER is easy to operate. However if you have a problem in operating the unit, follow the trouble-shooting guide below:

8.1 UPON POWER-UP THERE IS NO LED INDICATION.

Assure that the line cord is inserted properly.

8.2 EPROM DOES NOT VERIFY AFTER READ.

If, after you have read the EPROM into the SHOOTER'S internal RAM buffer and depressed the VER switch, the red LED goes on, the following problems may exist:

- Configurator plug inserted improperly or incorrect type. Remove the configurator plug and check for bent pins.
- EPROM inserted improperly. Make sure that the EPROM pin 1 is in the proper position.
- Bad EPROM, replace.

8.3 EPROM DOES NOT VERIFY AFTER PROGRAMMING

If, after the program cycle, the red LED goes on, it is an indication that the EPROM did not program properly. The following reasons could account for this program failure:

- Check the configurator plug for proper type, bent pins, and proper insertion.
- EPROM was not completely erased.
- EPROM is defective.
- EPROM is of incorrect type.
- EPROM inserted backward.
- VPP voltage failure. Check VPP voltage while in the program cycle.

9.Ø COMMUNICATIONS PROBLEMS

9.1 Terminal Mode:

9.1.1 MENU DOES NOT APPEAR ON THE SCREEN AFTER RESET.

Chances are your cabling is incorrect. Check to make sure that the following is true:

- SHOOTER'S rear RS-232 Connector pin 1 is the transmit line and should go to the receive pin of the terminal which is pin 3 for most terminals (refer to your terminal manual, not all terminals are the same).

- Pin 4 of SHOOTER is the receive pin. This pin should be connected to the transmit line of your terminal which should be pin 2.

- Pin 3 is the ground line and should be the only other line that is required to complete your connections.

You may be required to connect pins 4 to 5 and 6 to 2Ø on the terminal side, in order to force the handshake lines to the proper levels.

9.1.2 MENU APPEARS ON SCREEN BUT UNIT HANGS-UP.

This is an indication that the SHOOTER serial handshake line CTS (pin 5) is at low signal level.
This line is internally pulled up to +5V.

9.2 COMPUTER OPERATION

9.2.1 LED FLICKER ON DOWNLOAD

If you are downloading a file and you see the green and yellow LED flicker, the SHOOTER has not accepted the transfer (download) command. During a successful download operation the yellow (busy) LED is on continuously. This problem usually occurs because the user sends the "T" command immediately after the "I"or "M". Since the SHOOTER responds with *EOJ* <cr> lf after it receives the "I" (INTEL Format) or "M" (Motorola format) command, it will miss any characters sent to it while it is busy responding. Attempt sending the "I" seperately in the terminal mode.

After the "T" is received the programmer will ignore any character that is not a header character.
If no character is received within 7 seconds, the programmer will time-out.
If you do not wish to send the "I" command manually in terminal mode, you may send it automatically in two possible ways:

- The first method is to create a file with only the character "I" in the file, and another file with only the character "T". Now you must use command chain capability of your operating system to send the above files and your Hex file to the programmer with one simple command.

- The second method is to enter your Hex file into the editor and append the character "T" to the beginning of the file. If you are sending it in the Intel Hex format then you may also append an "I" to the file. To provide the timing required for the SHOOTER response you must type at least 15 "T's" after the "I".

Example: (using the CPM editor for a file named PROM.HEX)

```
A>ED PROM.HEX
                                        .
1=*5A APPEND 5 LINES TO THE TEXT BUFFER OF ED EDITOR
1=*ITTTTTTTTTTTTTTT Z  ;terminate insert with control Z
 =*E                   ;exit editor and save on disk
```

Now your PROM.HEX file has the proper command in the beginning of the file.

To send the file to the programmer:

```
PIP PUN:=PROM.HEX
```

This is assuming that your programmer is connected to the punch device.

Since using the editor to append the control characters to a file after every assembly can become quite cumbersome, you should try to use the SUBMIT command of CPM to send the three files automatically:

- I.HEX is a file containing the letter I only.
- T.Hex is a file containing the letter T only.
- You must create the above files using the editor.
- FILE.HEX is your Intel Hex file generated by your assembler.
- Create a file (submit file) with the following data:
- PIP PUN:=I.HEX,NUL:,T.HEX,$1.HEX
- Name the above file "XYZ.SUB"
- Now to send the file to the programmer type: SUBMIT XYZ FILE where FILE is the name of your INTEL Hex file. After a few seconds of disk access the SHOOTER busy light should go on solid for a few short minutes. After a successful download, the green LED should go on.

9.2.2 FOR NON CPM SYSTEMS

The above CPM procedure applies to most systems with the difference that certain command names and systems parameters may vary. In all cases, the important fact to remember is that if your computer sends any character other than the SHOOTER commands (with the exception of nulls), the SHOOTER will respond in half duplex with an "*". Suppose in sending the character "I" your computer actually precedes it with a control character. Then SHOOTER is going to miss the "I" because it is busy sending the "*". To get around this problem, simply send "IIII" instead of "I". This will allow several character time for response. If you encounter error problems during transmission, check the ground connection in your cable.

Make sure that your system is configured for:

ONE START BIT
8 DATA BITS
NO PARITY
2 STOP BITS

10.0 CALIBRATION CHECK

SHOOTER is tested and calibrated for the proper voltages prior to shipment at the factory. The user is not required to calibrate this unit. However, since even a small program voltage change can result in damage to an EPROM, it is suggested that you make periodic checks for the program voltages. Program voltage changes can happen only if certain component failures surface, or the configurator plug or an IC is not inserted properly.

The following tools are recommended for the voltage tests:

3 1/2 digit DVM
2 small clip leads
2 test pins

Voltage checks are done directly on the ZIF socket. Use a 1k 1 watt resister in the proper pin locations of the ZIF socket. The DVM clip leads are then attached to this resister for VPP voltage measurements.

One end of the resister is always used for the ground connection (pin 14 on the ZIF socket). The other end is inserted into the corresponding socket pin for the Program voltage.

10.1 25V PROGRAM VOLTAGE CHECK

This test will check the program voltage for all EPROM's requiring 25V program voltage:

1. Insert configurator plug #2716 in the CNF socket.

2. Insert a 1k resister into the ZIF socket between pins 23 and 14 (close ZIF socket handle for firm grip).

3. Attach the DVM ground clip to pin 14 and the second clip to pin 23 of the ZIF socket.

4. Reset the SHOOTER. When the yellow L.E.D. goes off the DVM should read below 2VDC (set DVM to 200VDC scale).

5. Depress the Program switch. The yellow L.E.D. should now be activated and the program voltage should read between 24.5 VDC and 25.5 VDC.

10.2 21VDC PROGRAM CHECK

This test is to check the voltage level for EPROM's requiring 21VDC program voltage (VPP) such as the 2732A EPROM.

1. Insert configurator plug #2732A in CNF socket.

2. Insert test pins in ZIF socket pins 14 and 22.

3. Set the DVM to the 200v scale and attach the ground clip to pin 14 of the ZIF socket and the second clip to pin 22 of the ZIF socket.

4. Press the Reset button. The DVM reading should be below 2VDC when the yellow (busy) led is off.

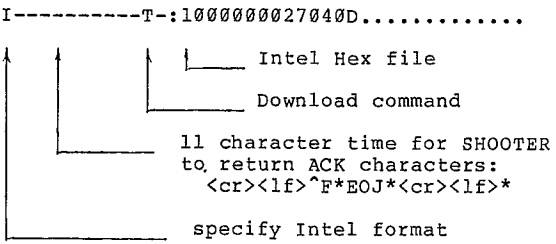
5. Depress PROG switch on front panel. Busy L.E.D. will indicate that the unit is in the program cycle. DVM must now read between 20.8 VDC and 21.8 VDC.

DO NOT HESITATE TO SEEK ASSISTANCE SHOULD YOU STILL EXPERIENCE ANY FURTHER PROBLEMS.

LOGICAL DEVICES, INC.
1321 N.W. 65 PLACE
FT. LAUDERDALE. FLORIDA 33309
CUSTOMER SERVICE: (305)974-0975

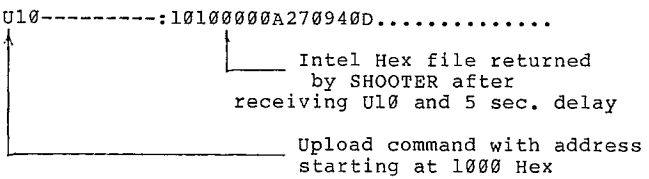
EXAMPLES OF UP-LOAD AND DOWN-LOAD

The following is the actual sequence of ASCII characters sent down to the programmer through the serial RS-232 port:



Note: the character time is needed because SHOOTER communicates in the half duplex mode: That is, it can only transmit or receive at one time. Therefore, while it is transmitting the acknowledge characters, it will not be able to see any characters transmitted to it by the computer. You can use the ASCII character "T" to fill in for character time. For further details refer to Section 7.0 and 7.1.
You will find an ASCII chart in APPENDIX B

FOR UPLOAD:



APPENDIX A

HEX FORMATS

FORMAT DESIGNATION	No. of ASCII CHARS.	CONTENT DESCRIPTION
Header Character	1	Always a colon ":" (HEX 3A)
Record length	2	A two-digit HEX value specifying the number of data bytes in the record. A record length of 0 indicates the last record in the file.
Starting Address	4	Four HEX digits representing the address of the memory location where the first Data Byte will be located.
Type of Record	2	00=normal data record 01=end of file record
Checksum	2	Two-digit HEX number representing two's complement of the sum, modulo 256 of the preceding eight bit bytes.

EXAMPLE :

```
:10000000027040DA5159327D7D5BF8C043A9A8FA83E

: 10 0000 00 27 04 0D A5 15 93 27 D7 D5 BF 8C 04 3A 9A 8F A8 3E
- -- ---- -- ------------------------------------------------ --
| |  |    |                        |                        |
| |  |    |__Type of Record        Data                     Checksum__|
| |  |_______Starting Address
| |__________Record Length
|____________Header
```

AN EXAMPLE OF A 1K BYTE INTEL OBJECT MODULE :

```
:10000000027040DA5159327D7D5BF8C043A9A8FA83E
:10001000A999CF54170A7221347902346B961804C7
:1000200035994F14F59ACF088A209A7F0214EF363B
:1000300032E7D99625D5ABBF43B53435144234428C1
:10004000043C65C589DF9AA08A802702A8A9AA8AEC
:10005000902 3C03A86540509379 2B6850AD2629534
:1000600004B676575455D5535FAF99CF9ABF03B3B3
:10007000967 4AB930304967BBB029303F5C59682FB
:10008000646060796A9C5AE54CE47AABC60541BECC3
:10009000 8D54708A080A729EBE4026A004A226A231
:1000A000BE80D5FBC532CD24E607C65F03FE96B2FF
:1000B00064BF030896069A8F99CF54178A080A726C
:1000C000C3243BB6CB26C92487442824062 33A54AC
:1000D000872310AB54A52754785 4B1175412EECD92
:1000E000233A54872754A52301547823EF4412C59B
:1000F00018F896F519C58AA09AEFF802F943B03AB4
:100100009AEF802308938948 5417997F541727C57D
:10011000A98A0814F59ACF08AF8A209A7F54FF1451
:10012000EF3624E7D9961593BF8509B2312306240B
:1001300033231554891FFFE39633939A8FF8B65FE4
:1001400002F93A43103A3A3A8002997F238F3923D1
```

```
:100150005C541D8940997F8980346B8A20963BA826
:10016000A93479AF54FF346B9661938A8018F8965E
:10017000781923043677E7D983F802F943803A3AAD
:10018000BAB080089A4F83C514F59A6F800299B798
:10019000BA88541B9AF718F8968819F972062488E9
:1001A000BAC8D0DBBCCAD5E089060953067703A0DC
:1001B000B3890609530677O3A4B334BCD5BCF0EC6D
:1001C000BFECC1BC1BECC59334CAD5BCBCECCD9311
:1001D000D5BC29ECD3D5BC2AECD893D5BC11ECDE28
:1001E000D5BC10ECE39323535487BB002331540850
:1001F000EEE623535487233954872303547827S436
:10020000782754782JFC44125487231354FA54A5B6
:1002100054B154785470935419541D238362551665
:10022000215423162744238384A09AEFF802F94326
:10023000303A800299B79ADF541B8A2018F8962822
:1002400019F93792282406EC530406864BD505BCD1
:10025000DC16477651D5B515BA01B90034B19734DB
:10026000A8FAF72A49F66C866AA9445F34B1F99373
:10027000230D5489230A4489C5AD47547EFD530F8D
:10028000033057170287075687D5A8BA019799FE2A
:1002900008344A8F69FFAF72A58C68E890144918936
:1002A0000134A824A85478C5F96A54ADF854FA4426
:1002B0007814F5BF109A6F0854FA547814EFEFB51C
:1002C000FB3793544BC554D047AE54CE4E93544B4A
:1002D00003D0F2E3975792DBB2DB9303EFF2E30331
:1002E000FAF2E6892093031093C554EF530FA954F3
:1002F000C354FAA89354C39A6F02C52B6B2B93086F
:10030000DFC605892093D555FB3291544BD35396C4
:1003100008B5544B530FC5C60BBB00D503F7962175
:1003200095S4C354FA03FDAD54E9D5FDC63814F510
:10033000643414EF54F5ED3254C3C537DBD5740182
:1003400095B60B93111A0D0A6528582961ODODOA8F
:10035000496E74656C0D0A4D6F746F726F6C610D30
:1003600000A50726F6772616D0D0A5472616E736626
:1003700065720D0A55706C6F61640D0A5665572697D
:1003800066790D0A2A000D0A2A454F4A2A0D0A2AC3
:1003900000544BD33A969185S4C3ADC5AB54E9D5BF
:1003A00054C354FAC6AA95FDC6B414F564B014EF4C
:1003B00054F5EDAE54C3C56BD5740195B691932336
:1003C00020548754E9D55470C5230F59A9D554A793
:1003D000233D546714F59A6F0854782320548754SA
:1003E0004B03E0C6FC0313C6BEF999DF54C5AC0944
:1003F000B2C6FC9008DCC6FC233F548714EF64C6E9
:00000001FF
```

MOTOROLA LOAD MODULE FORMAT

FORMAT DESIGNATION	No. of ASCII CHARS.	CONTENT DESCRIPTION
Start of Record	1	Always an "S" (HEX 53)
Type of Record	1	0-header record (sometimes not used) 1-normal data record 9-end of file record
Byte Count	2	A two-digit HEX value specifying the number of data bytes in the record, including the address and the check-sum. Each Byte is represented as two HEX characters.
Address	4	Four HEX digits representing the address of the memory location where the first Data Byte will be located.
Checksum	2	Two-digit HEX number representing one's complement of the sum, modulo 256, of the Byte Count, Address, and the Data Bytes.

EXAMPLE :

S113000027040DA5159327D7D5BF8C043A9A8FA83A (normal data record)

```
S 1 13 0000 27 04 0D A5 15 93 27 D7 D5 BF 8C 04 3A 9A 8F A8 3A
- - -- ---- ------------------------------------------------- --
| | |  |                          |                            |
| | |  |__Starting Address       Data            Checksum__|
| | |______Byte Count
| |________Type of Record
|__________Start of Record (header character)
```

AN EXAMPLE OF A 1K BYTE MOTOROLA OBJECT MODULE :

```
S113000027040DA5159327D7D5BF8C043A9A8FA83A
S1130010A999CF54170A7221347902346B961804C3
S113002035994F14F59ACF088A209A7F0214EF3637
S113003032E7D99625D5ABBF43B534351442342BBD
S113004043C65C589DF9AA08A802702A8A9AA8AE8
S113005090023C03A865405093792B6850AD2629530
S11300604B676575455D5535FAF99CF9ABF03B3AF
S113007096 74AB930304967BBB029303F5C59682F7
S11300806406 0796A9C5AE54CE47AABC60541BECBF
S113009080D54708A080A729EBE4026A004A226A22D
S11300A0BE80D5FBC532CD24E607C65F03FE96B2FB
S11300B064BF030896069A8F99CF54178A080A7268
S11300C0C3243BB6CB26C92487442824062 33A54A8
S11300D0872310AB54A527547854B1175412EECD8E
S11300E0233A54872754A52301547823EF4412C597
S11300F018F896F519C58AA09AEFF802F943B03AB0
S11301009AEF80230893894854 17997F541727C579
S1130110A98A0814F59ACF08AF8A209A7F54FF144D
S1130120EF3624E7D9961593BF8509B23123062407
S11301303323155489 1FFFE39633939A8FF8B65FE0
S113014002F93A43103A3A3A8002997F238F3923CD
S11301505C541D8940997F8980346B8A20963BA822
S1130160A93479AF54FF346B9661938A8018F8965A
S11301707819230436 77E7D983F802F943803A3AA9
S113018 08AB080089A4F83C514F59A6F800299B794
S113019 08A88541B9AF718F8968819F972062488E5
S11301A0BAC8D0DBBCCAD5E08906095306 7703A0D8
S11301B0B389060953067703A4B334BCD5BCF0EC69
S11301C0BFECC1BC1BECC59334CAD5BCBCECCD930D
S11301D0D5BC29ECD3D5BC2AECD893D5BC11ECDE24
S11301E0D5BC10ECE39323535487BB0023315 4084C
S11301F0EEE62353548723395487230354782 75432
S1130200782754782 3FC44125487231354FA54A5B2
S1130210 54B154785470935419541D238362551661
S11302202154231627442383 8AA09AEFF802F94322
S1130230303A800299B79ADF541B8A2018F896281E
S113024019F93792282406EC530406864BD505BCCD
S1130250DC164776 51D5B515BA01B90034B19734D7
S11302 60A8FAF72A49F66C866AA9445F34B1F9936F
S1130270230D5489230A4489C5AD47547EFD530F89
S11302800 3305717D287075687D5A8BA019799FE26
```

```
S11302900834A8F69FFAF72A58C68E8901449189 32
S11302A00134A824A85478C5F96A54ADF854FA4422
S11302B07814F5BF109A6F0854FA547814EFEFB518
S11302C0FB3793544BC554D047AE54CE4E93544B46
S11302D003D0F2E3975792DBB2DB9303EFF2E3032D
S11302E0FAF2E689209303109 3C554EF530FA954EF
S11302F0C354FAA89354C39A6F02C52B6B2B93086B
S1130300DFC605892093D555FB3291544BD35396C0
S1130310 0B85544B530FC5C60BBB00D503F7962171
S1130320 9554C354FA03FDAD54E9D5FDC63814F50C
S1130330643414EF54F5ED3254C3C537DBD574017E
S1130340 95B60B93111A0D0A65285829616D0D0A8B
S1130350496E74656C0D0A4D6F746F726F6C610D2C
S1130360 0A50726F6772616D0D0A5472616E736622
S1130370657200A55706C6F61640D0A5665726979
S1130380667900A2A000D0A2A454F4A2A0D0A2ABF
S1130390 0544BD33A96918554C3ADC5AB54E9D5BB
S11303A054C354FAC6AA95FDC6B414F564B014EF48
S11303B054F5EDAE54C3C56BD5740195B691932332
S11303C020548754E9D55470C5230F59A9D554A78F
S11303D0233D548714F59A6F085478232054875486
S11303E04B03E0C6FC0313C6BEF999DF54C5AC0940
S11303F0B2C6FC9008DCC6FC233F548714EF64C6E5
S9030000FC
```

S9030000FC (end of file record)

```
        TITLE   'UNLOAD ver 2.1 - Create HEX file from COM file'
;       (revised 05/20/81)
;
;--->Needs MAC and SEQIO.LIB to assemble<---
;
;Originally from CPMUG 29.23
;
;05/20/81 Modified for 32 bytes/record instead of 16, for less
;         overhead in the .HEX output file, by Dav Holle.
;
;11/07/80 Modified to default to 100H, increase size of buffers,
;         add signon message.  By Keith Petersen, W8SDZ
;
;To use, type: UNLOAD <FILENAME> <ADDR>
;
;Where: <FILENAME>.COM is the source file
;       <FILENAME>.HEX will be the output file
;       <ADDR> is the optional start address in hex (default=100)
;
        ORG     100H
;
        MACLIB  SEQIO   ;DEFINE MACRO LIBRARY USED
;
        LHLD    6       ;GET BASE OF BDOS
        DCX     H       ;BACK OFF ONE BYTE
        SPHL            ;SET STACK THERE
        CALL    SIGNON
        DB      CR,LF,'UNLOAD ver 2.1',CR,LF,'$'
        DB      '05/20/81' ;REVISION DATE (doesn't print)
;
SIGNON: POP     D       ;GET MSG ADR
        MVI     C,@MSG  ;PRINT IT
        CALL    @BDOS
        LXI     H,100H  ;DEFAULT UNLOAD ADRS
        LXI     D,FCB2+1
        LDAX    D       ;GET OPTION ADRS
        CPI     ' '     ;ANY GIVEN?
        JZ      INITFL  ;NO, DEFAULT TO 100H
        LXI     H,0
        MVI     B,0
;
ADRLUP: LDAX    D
        INX     D
        SUI     '0'
        JC      INITFL
        CPI     10
        JC      ADDNIB
        SUI     7
        JC      INITFL
        CPI     16
        JNC     INITFL
;
ADDNIB: DAD     H
```

```
        DAD     H
        DAD     H
        MOV     C,A
        DAD     B
        JMP     ADRLUP

;
INITFL: PUSH    H
        FILE    INFILE.SOURCE.,1,COM,2048
        FILE    OUTFILE,OUTPUT,,1,HEX,2048
        POP     H

;
ADRDON: SHLD    LODADR

;
UNLOOP: GET     SOURCE
        JZ      GEOF
        PUSH    PSW
        MVI     A,':'
        PUT     OUTPUT
        XRA     A
        STA     CHEKS
        MVI     A,32    ;was 16
        CALL    PUTBYTE
        LDA     LODADR+1
        CALL    PUTBYTE
        LDA     LODADR
        CALL    PUTBYTE
        XRA     A
        CALL    PUTBYTE
        POP     PSW
        MVI     B,32    ;was 16

;
LINLUP: PUSH    B
        CALL    PUTBYTE
        POP     B
        DCR     B
        JZ      NEXTL
        GET     SOURCE
        JMP     LINLUP

;
PUTBYTE:MOV     C,A
        LDA     CHEKS
        SUB     C
        STA     CHEKS
        MOV     A,C
        RRC
        RRC
        RRC
        RRC
        CALL    PUTNIB
        MOV     A,C

;
PUTNIB: ANI     0FH
        ADI     '0'
        CPI     '9'+1
        JC      PUTNB1
        ADI     7

;
PUTNB1: PUSH    B
        PUT     OUTPUT
        POP     B
        RET

;
NEXTL:  LDA     CHEKS
        CALL    PUTBYTE
        MVI     A,CR
```

```
        PUT     OUTPUT
        MVI     A,LF
        PUT     OUTPUT
        LHLD    LODADR
        LXI     D,32    ;was 16
        DAD     D
        JMP     ADRDON
;
GEOF:   MVI     A,':'
        PUT     OUTPUT
        MVI     B,5
;
GEOF1:  XRA     A
        PUSH    B
        CALL    PUTBYTE
        POP     B
        DCR     B
        JNZ     GEOF1
        MVI     A,CR
        PUT     OUTPUT
        MVI     A,LF
        PUT     OUTPUT
        FINIS   OUTPUT
        LXI     D,DMSG
        MVI     C,9MSG
        CALL    5BDOS
        JMP     0
;
DMSG:   DB      'DONE',CR,LF,'$'
;
LODADR: DS      2
CHEKS:  DS      1
FCB2    EQU     5CH
;
BUFFERS EQU     $       ;INPUT/OUTPUT BUFFERS GO HERE
;
        END
```

8080 Program to Generate Intel Hex Format

```
        READ COMMAND    READ INTEL HEX FORMAT TAPE (W/BIAS IF DESIRED)
READ:   CALL    EXPRI   ;GET BIAS INPUT IF DESIRED
        POP       H     ;PLACE BIAS INTO HL (IS 0000 OF NO BIAS)
REDO:   CALL      RI    ;READ FROM LOGICAL TAPE READER DEVICE
        RC              ;RETURN IF NO BYTE IS AVAILABLE
        ANI      7FH    ;MASK OFF POSSIBLE BITS
        SUJ      ':'    ;CLUMSY COMPARISON FOR START LINE
        JNZ      REDO   ;LOOP TIL START OF LINE DELIMINATOR FOUND
        MOV       D,A   ;A MUST BE ZERO, SO USE IT TO CLEAR D RFG
        PUSH      H     ;SAVE BIAS VALUE ON STACK
        CALL    BYTE    ;GET FIRST DATA BYTE
        JZ      RED2    ;IF ZERO, INDICATES LAST LINE AND LOAD ADDRES
        MOV       F,A   ;LOAD NUMBER OF DATA BYTES IN LINE INDICATOR
        CALL    BYTE    ;GET FIRST ADDRESS BYTE
        MOV       B,A   ;SAVE HIGH ORDER ADDRESS IN B
        CALL    BYTE    ;GET LOWER ADDRESS BYTE
        MOV       C,A   ;SAVE LOW ORDER ADDRESS IN C
        DAD       B     ;ADD BIAS TO LOADING ADDRESS FROM LINE
        CALL    BYTE    ;JUST READ TERMINATOR BYTE
RED1:   CALL    BYTE    ;READ DATA BYTE
        MOV       M,A   ;STORE DATA BYTE AT LOAD ADDR+BIAS
        INX       H     ;BUMP STORAGE POINTER
        DCR       F     ;DECREMENT BYTE COUNT FOR LINE
        JNZ     RED1    ;LOOP TILL BYTE COUNT IS EXHAUSTED
        CALL    BYTE    ;READ CHECK BYTE AND DO FINAL SUM
        POP       H     ;RESTORE BIAS FOR NEXT LINE
        JZ      REDO    ;JUMP FOR NEXT LINE IF CHECKSUM=0
        STC             ;SET THE CARRY BIT ON AS AN ERROR INDICATOR
        RFT
RED2:   CALL    BYTE    ;GET TWO ADDRESS BYTES AND BRANCH THERE
        MOV       H,A
        CALL    BYTE
        POP       B     ;REMOVE BIAS VALUE FROM STACK
        MOV       L,A   ;CHECK THE EXECUTION ADDRESS FOR ZERO
        URA       H
        RZ              ;RETURN TO MONITOR ON A ZERO EXECUTION ADDR
        PCH1            ;GO TO EXECUTION ADDRESS
BYTE:   CALL    RNBBL   ;READ UPPER NIBBLE OF DATA BYTE
        RLC             ;SHIFT TO UPPER 4 BITS
        RLC
        RLC
        RLC
        MOV       C,A   ;SAVE MS NIBBLE IN C
        CALL    RNBBL   ;READ SECOND NIBBLE
        ORA       C     ;COMBINE NIBBLES TO FORM HEX BYTE
        MOV       C,A   ;SAVE BYTE IN C
        ADD       D     ;ADD TO CHECK SUM
        MOV       D,A   ;SAVE NEW CHECK SUM
        MOV       A,C   ;RESTORE DATA BYTE TO ACCUM
```

```
RNBB1:  CALL    RJ      ;DO READER INPUT
        JC      RNBER   ;IF CARRY SET, ERROR
        ANJ     ZFH     ;MASK OFF PARITY BIT
        CALL    NIBBL   ;READ NIBBLE FROM ASCII HEX BYTE
        JC      RNBR    ;JUMP ON ERROR
        RET
RNBFR:  POP     H       ;RETURN TO MONITOR ON ERROR
        POP H
        POP H
        RET
```

```
C11 17  01A1  LOAD1   LBSR   ECHON    INBUT 8 BIT BYTE WITH NO ECHO
C14 81  53    LOAD2   CMPA   #'S      IS IT S, START CHAR?
C16 26  F9            BNE    LOAD1    NO, DISCARD AND GET NEXT CHAR
C18 17  019A          LBSR   ECHON
C1B 81  39            CMPA   #'9      IS 9, END OF FILE CHAR?
C1D 27  3D            BEQ    LOAD21   IS SO, EXIT LOAD
C1F 81  31            CMPA   #'1      IS 1, FILE LOAD CHAR?
C21 26  F1            BNE    LOAD2    IF NOT, LOOK FOR START CHAR
C23 17  0116          LBSR   BYTE     INPUT BYTE CNT
C26 34  02            PSHS   A        PUSH COUNT ON STK
C28 29  26            BVS    LODERR   (V)C-CODE SET, ILLEGAL HEX
C2A 17  00FE          LBSR   IN1ADR   INPUT LOAD ADDR
C2D 29  21            BVS    LODERR   (V) C-CODE SET, ADDR NOT HEX
C2F 34  10            PSHS   X        PUSH ADDR ON STK
C31 E6  E0            LDB    ,S+      LOAD MSB OF ADDR AS CKSUM BYTE
C33 EB  E0            ADDB   ,S+      ADD LSB OF ADDR TO CKSUM
C35 EB  E4            ADDB   ,S       ADD BYTE COUNT BYTE TO CKSUM
C37 6A  E4            DEC    ,S       $FC37 DEC BYTE COUNT 2 TO BYPASS
C39 6A  E4            DEC    ,S       ADDRESS BYTES
C3B 34  04    LOAD10  PSHS   B        PUSH CKSUM ON STK
C3D 17  00FC          LBSR   BYTE     INPUT DATA BYTE - 2 HEX CHARS
C40 35  04            PULS   B        POP CKSUM FROM STK
C42 29  0C            BVS    LODERR   (V) SET, DATA BYTE NOT HEX
C44 34  02            PSHS   A        PUSH DATA BYTE ON STK
C46 EB  E0            ADDB   ,S+      ADD DATA TO CKSUM, AUTO INC STK
C48 6A  E4            DEC    ,S       DEC BYTE COUNT 1
C4A 27  05            BEQ    LOAD16   IF BYTE COUNT ZERO, CHECK CKSUM
C4C A7  80            STA    ,X+      SAVE DATA BYTE IN MEM
C4E 20  EB            BRA    LOAD10   GET NEXT DATA BYTE
C50 5F          LODERR CLRB            ERR COND, ZERO CKSUM
C51 35  02      LOAD16 PULS   A        ADJUST STK (REMOVE BYTE CNT)
C53 C1  FF            CMPB   #$FF     CKSUM OK?
C55 27  B2            BEQ    LOAD     IF SO, LOAD NEXT LINE
C57 86  3F            RDA    #'?      LOAD "?", ERR INDICATOR
C59 17  0183          LBSR   OUTCH    OUTPUT TO TERM
C5C 73  DFE2  LOAD21  COM    ECHO     TURN ECHO ON
C5F 86  13            LDA    #$13     $FC5F LOAD "DC3" CASS READ OFF COD
C61 16  017B          LBRA   OUTCH    OUTPUT IT
                      *
                      * "P" PUNCH MIKBUG TAPE
                      *
C64 6F  E2    PUNCH   CLR    ,-S      CLEAR RES BYTE ON STK
C66 17  00B7          LBSR   IN2ADR   GET BEG & END ADDR
C69 34  30            PSHS   X,Y      SAVE ADDR'S ON STK
C6B 29  4A            BVS    PUNEXT   (V) C-CODE SET, EXIT PUNCH
C6D AC  62            CMPX   2,S      CMP BEG TO END ADDR
C6F 25  46            BCS    PUNEXT   IF BEG>END, EXIT PUNCH
C71 30  01            LEAX   1,X      INC END ADDR
C73 AF  E4            STX    ,S       STORE END ADDR ON STK
C75 86  12            LDA    #$12     LOAD 'DC2' PUNCH ON CODE
```

```
FC77 17   0165          LBSR   OUTCH    OUTPUT TO TERM
FC7A EC   E4    PUNCH2  LDD    ,S       LOAD END ADDR IN D REG
FC7C A3   62            SUBD   2,S      SUB BEG FROM END
FC7E 27   06            BEQ    PUNCH3   SAME, PUNCH 32 BYTES OF DEFAULT
FC80 1083 0020          CMPD   #$20     LESS THAN 32 BYTES?
FC84 23   02            BLS    PUNCH4   PUNCH THAT MANY BYTES
FC86 C6   20    PUNCH3  LDB    #$20     LOAD BYTE CNT OF 32
FC88 E7   64    PUNCH4  STB    4,S      STORE ON STK AS BYTE CNT
FC8A 8E   FEDC          LDX    #MSG20   POINT TO MSG "S1"
FC8D 17   010E          LSBR   PSTRNG   PRINT MSG
FC90 CB   03            ADDB   #3       ADD 3 BYTES TO BYTE CNT
FC92 1F   98            TFR    B,A      GET BYTE CNT I A-REG TO PUNCH
FC94 17   00DB          LBSR   OUT2H    OUTPUT BYTE COUNT
FC97 AE   62            LDX    2,S      LOAD BEG ADDR
FC99 17   00CE          LBSR   OUT4H    PUNCH ADDR
FC9C EB   62            ADDB   2,S      ADD ADDR MSB TO CKSUM
FC9E EB   63            ADDB   3,S      ADD ADDR LSB TO CKSUM
FCA0 EB   84    PUNCHL  ADDB   ,X       ADD DATA BYTE TO CKSUM
FCA2 A6   80            LDA    ,X+      LOAD DATA BYTE TO PUNCH
FCA4 17   00CB          LBSR   OUT2H    OUTPUT DATA BYTE
FCA7 6A   64            DEC    4,S      DEC BYTE CNT
FCA9 26   F5            BNE    PUNCHL   NOT DONE, PUNCH NEXT BYTE
FCAB 53                 COMB             1'S COMPLIMENT CKSUM BYTE
FCAC 1F   98            TFR    B,A      PUT IN A REG TO PUNCH
FCAE 17   00C1          LBSR   OUT2H    OUTPUT CKSUM BYTE
FCB1 AF   62            STX    2,S      SAVE X REG IN STK AS NEW PUNCH ADD
FCB3 AC   E4            CMPX   ,S       COMPARE TO END ADDR
FCB5 26   C3            BNE    PUNCH2   $FCB5 PUNCH NOT DONE, CONT
FCB7 86   14    PUNEXT  LDA    #$14     LOAD "DC4" PUNCH OFF CODE
FCB9 17   0123          LBSR   OUTCH    OUTPUT IT
FCBC 32   65            LEAS   5,S      READJ STK PTR
FCBE 39                 RTS
```

```
10 'PROGRAM NAME IS                    "RELOC"
20 '
30 'THIS PROGRAM RELOCATES THE CODE IN AN EXISTING INTEL HEX FILE NAMED
40 ' "FILENAME.HEX". IT STARTS THE CODE AT THE DESIRED STARTING ADDRESS,
50 'RECALCULATES THE CHECKSUMS AND OUTPUTS THE MODIFIED FILE WITH THE NAME
60 ' "FILENAME.REL".  THE PROGRAM ALSO KEEPS A RUNNING TOTAL OF
70 'THE NUMBER OF BYTES OF CODE THAT ARE BEING TRANSFERRED TO THE
80 'NEW FILE.
90 INPUT"NAME OF HEX FILE TO RELOCATE";NA$
100 INPUT"DESIRED STARTING ADDRESS";SA$
110 SA=VAL("&H"+SA$):CB=0
120 PRINT
130 PRINT"NUMBER OF BYTES WRITTEN FOLLOWS"
140 NAMIN$=NA$+".HEX"
150 OPEN"I",#1,NAMIN$
160 NAMOUT$=NA$+".REL"
170 OPEN"O",#2,NAMOUT$
180 IF EOF(1) THEN 400:'          STOP ON END OF FILE
190 LINE INPUT#1,A$:'             GET RECORD
200 X$=MID$(A$,2,2)
210 NTOT=VAL("&H"+X$):'          NUMBER OF BYTES IN RECORD
220 IF NTOT<>0 THEN 250:'        CHECK IF LAST RECORD
230 PRINT#2,A$:'                 OUTPUT LAST RECORD
240 GOTO 400:'                   STOP
250 CS=10+2*NTOT:'                LOCATION OF CHECKSUM CHARACTERS
260 NC=LEN(SA$):'                 NUMBER OF CHARACTERS IN ADDRESS
270 IF NC=4 THEN SAA$=SA$
280 IF NC=3 THEN SAA$="0"+SA$
290 IF NC=2 THEN SAA$="00"+SA$
300 IF NC=1 THEN SAA$="000"+SA$
310 MID$(A$,4,4)=SAA$:'           SET NEW STARTING ADDRESS
320 GOSUB 440:'                CALL CHECKSUM
330 MID$(A$,CS,2)=CS$:'           SET NEW CHECKSUM
340 PRINT#2,A$:'                 OUTPUT MODIFIED RECORD
350 SA=SA+NTOT:'                 INCREMENT STARTING ADDRESS
360 CB=CB+NTOT:'                CUMULATIVE NUMBER OF BYTES
370 PRINT CB
380 SA$=HEX$(SA)
390 GOTO 180:'                    DO IT AGAIN
400 CLOSE#1
410 CLOSE#2
420 PRINT"MODIFIED FILE IS CALLED,",NAMOUT$
430 STOP
440 '.................SUBROUTINE CHECKSUM.........................
450 SUM=0:X=0
460 FOR J=2 TO 8+2*NTOT  STEP 2
470 N$=MID$(A$,J,2)
480 GOSUB 560:'               CALL HEXTODEC
490 SUM=SUM+X
500 IF SUM>256 THEN SUM=SUM-256
510 NEXT J
520 CS$=HEX$(256-SUM)
530 LC=LEN(CS$)
540 IF LC=1 THEN CS$="0"+CS$
550 RETURN
560 '.................SUBROUTINE HEXTODEC.......................
570 X=VAL("&H"+N$)
580 IF X<0 THEN X=X+2^16
590 RETURN
```

Relocated next time

FILE.REL

Starts 0000H

```
:08000000E6F6003130ECFFFCD4
:08000800FDD61062D61057D698
:0800100010624C10D610694A81
:08001800FBD610E2D610AC8DFE
:0800200010 3E7C122C0092201E
:0800280A0E07AFAAF31300CC0
:0800300181C00AFE602306C61
:0800380026AFEE602106C0FE3
:0800400006AFE3C00D610627C30
:0800480006D610847AFBAF59C3
:0800500002822E82300223928D
:0800580020 5E590282 2E56E2DF
:0800600007FA0E0823012239220
:0800680020A0E0FD10A8823089
:0800700 3E9230A0E05EAF7C7F
:0800780006D6106282 80A0E0B0
:0800800082 90A0E082A0BC0404
:08008800FD10C1EFC0EAC0E960
:0800900 0C0E8BAF499F0D610A3
:0800980 0D869F0D610D8A0E0D1
:0800A0007ADAAF76FA106D1058
:0800A800D8E6FA00AFBC0CE63B
:0800B000F0FFD610D8BAF8E603
:0600B800F000D610D8AFE5
:00000001FF
```

original hex file

FILE.HEX

starts 1035 H

```
:08103500E6F6003130ECFFFC8F
:08103D00FDD61062D61057D653
:0810450010624C10D610694A3C
:08104D00FBD610E2D610AC8DB9
:0810550103E7C122C009220D9
:08105D00A0E07AFAAF31300C7B
:0810650181C00AFE602306C1C
:08106D00026AFEE602106C0F9E
:0810750006AFE5C00D610627CEB
:08107D0006D610847AFBAF597E
:0810850002822E823002239248
:08108D00205E5902822E56E29A
:0810950007FA0E0823012239 2DB
:08109D0020A0E0FD10A8823044
:0810A5003E9230A0E05EAF7C3A
:0810AD0006D610628280A0E06B
:0810B5008290A0E082A0BC04BF
:0810BD00FD10C1EFC0EAC0E91B
:0810C500C0E8BAF499F0D6105E
:0810CD00D889F0D610D8A0E08C
:0810D5007ADAAF76FA106D1013
:0810DD00D8E6FA00AFBC0CE6F6
:0810E500F0FFD610D8BAF8E6BE
:0610ED00F000D610D8AFA0
:00000001FF
```

APPENDIX B

SOFTWARE DRIVERS

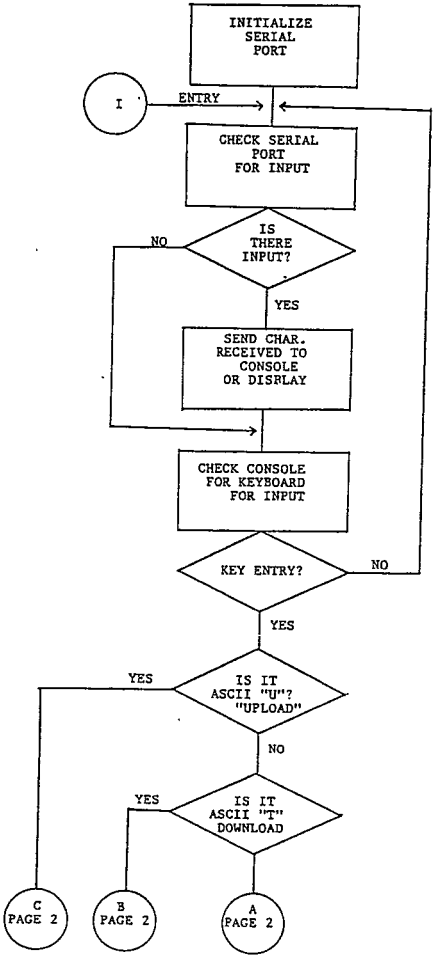
ASCII CODES (HEXIDECIMAL)

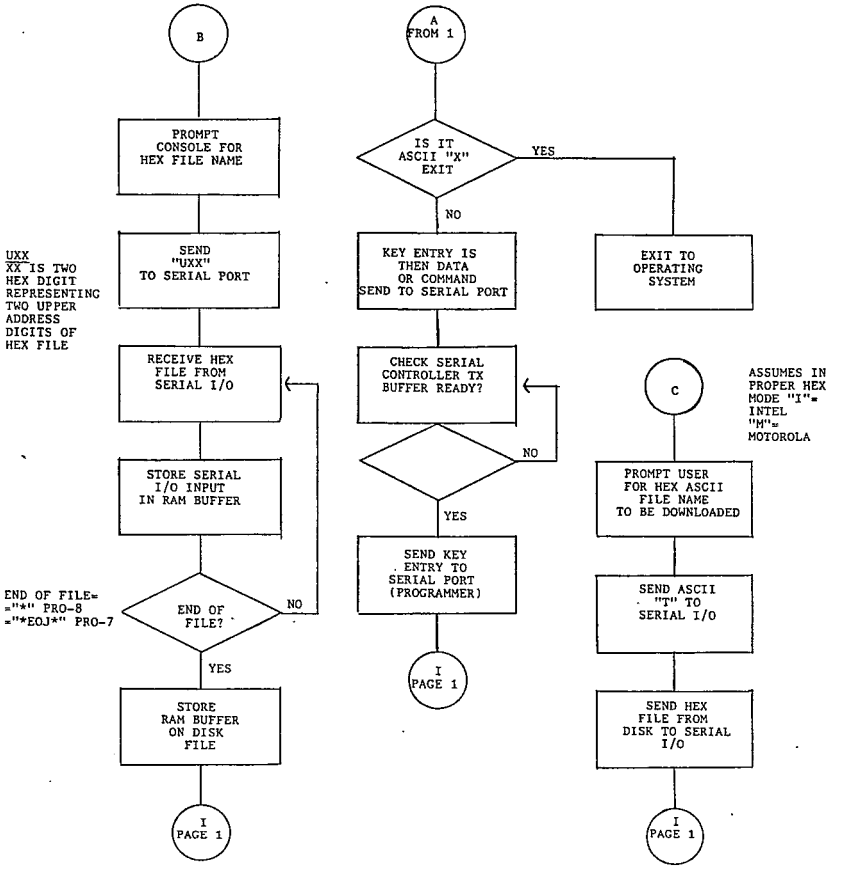
				B7	0	0	0	0	0	0	0	0
				B6	0	0	0	0	1	1	1	1
				B5	0	0	1	1	0	0	1	1
				B4	0	1	0	1	0	1	0	1
BINARY				HEX								
B3	B2	B1	B0	$	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	\	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STK	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	\	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	A	LF	SUB	*	:	J	Z	j	z
1	0	1	1	B	VT	ESC	+	;	K	[	k	{
1	1	0	0	C	FF	FS	,	<	L	\	l	|
1	1	0	1	D	CR	GS	-	=	M	]	m	}
1	1	1	0	E	SO	RS	.	>	N	^	n	~
1	1	1	1	F	SI	US	/	?	O	_	o	RUBOUT

NUL	NULL
SOH	START OF HEADING
STX	START OF TEXT
ETX	END OF TEXT
EOT	END OF TRANSMISSION
ENQ	ENQUIRY
ACK	ACKNOWLEDGE
BEL	BELL
BS	BACK SPACE
HT	HORIZONTAL TABULATION
LF	LINE FEED
VT	VERTICAL TABULATION
FF	FORM FEED
CR	CARRIAGE RETURN
SO	SHIFT OUT
SI	SHIFT IN

DLE	DATA LINK ESCAPE
DC1	DEVICE CONTROL 1
DC2	DEVICE CONTROL 2
DC3	DEVICE CONTROL 3
DC4	DEVICE CONTROL 4
NAK	NEGATIVE ACKNOWLEDGE
SYN	SYNCHRONOUS IDLE
ETB	END OF TRANSMISSION
CAN	CANCEL
EM	END OF MEMORY
SUB	SUBSTITUTE
ESC	ESCAPE
FS	FILE SEPARATOR
GS	GROUP SEPARATOR
RS	RECORD SEPARATOR
US	UNIT SEPARATOR

GENERALIZED SOFTWARE DRIVER FLOW CHART




B
PROMPT CONSOLE FOR HEX FILE NAME
SEND "UXX" TO SERIAL PORT
UXX XX IS TWO HEX DIGIT REPRESENTING TWO UPPER ADDRESS DIGITS OF HEX FILE
RECEIVE HEX FILE FROM SERIAL I/O
STORE SERIAL I/O INPUT IN RAM BUFFER
END OF FILE= ="*" PRO-8 ="*EOJ*" PRO-7
END OF FILE?
NO
YES
STORE RAM BUFFER ON DISK FILE
I PAGE 1
A FROM 1
IS IT ASCII "X" EXIT
YES
NO
KEY ENTRY IS THEN DATA OR COMMAND SEND TO SERIAL PORT
EXIT TO OPERATING SYSTEM
CHECK SERIAL CONTROLLER TX BUFFER READY?
NO
YES
SEND KEY ENTRY TO SERIAL PORT (PROGRAMMER)
I PAGE 1
C
ASSUMES IN PROPER HEX MODE "I"= INTEL "M"= MOTOROLA
PROMPT USER FOR HEX ASCII FILE NAME TO BE DOWNLOADED
SEND ASCII "T" TO SERIAL I/O
SEND HEX FILE FROM DISK TO SERIAL I/O
I PAGE 1

TRS-80 COLOR COMPUTER

DOWNLOAD PROGRAM

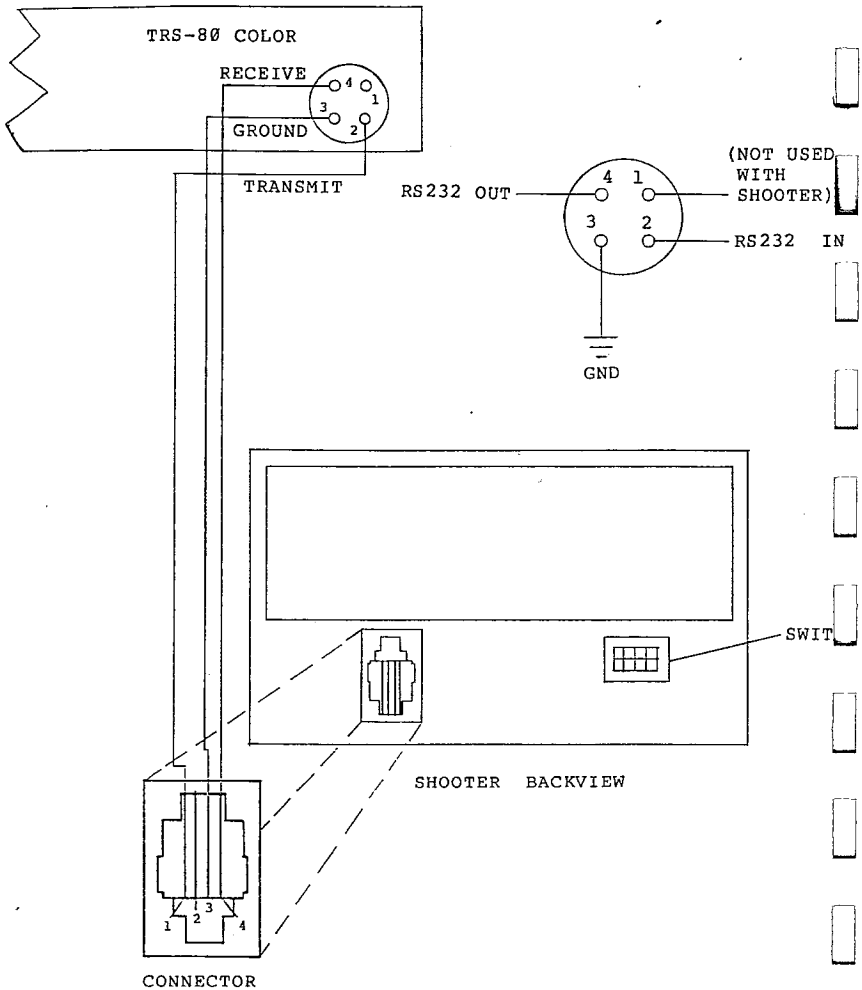
This basic program will download from RAM (source) addresses as specified with selectable destination address. The "Device Command Line" as entered is sent in front of the Motorola Format File (add "T" for DOWNLOAD) to allow control of the PROM Programmer. The file ends with "S9".

The program as supplied should be compatible with extended or non-extended TRS-80 Color Systems. The comment lines show changes which will make the program faster with extended Basic.

To use your Color Computer as a terminal. You can utilize standard terminal programs such as the "Super Color" provided by Microware.

```
10 ' PROGRAM TO DOWNLOAD MEMORY FROM COLOR COMPUTER
20 ' MEMORY IN MOTOROLA HEXADECIMAL FORMAT
30 ' THIS PROGRAM IS COMPATIBLE WITH EXTENDED
40 ' OR NON-EXTENDED COLOR BASIC
50 INPUT "BAUD RATE = ";V
60 V=INT(55500/V-4.5)
70 V1=INT(V/256):POKE 149,V1
80 V=V-256*V1:POKE 150,V
85 PRINT #-2, "M"
90 INPUT "SOURCE" STARTING ADDRESS (HEX)= ";V$
100 GOSUB 270:SS=V
110 INPUT "SOURCE ENDING ADDRESS (HEX)= ";V$
120 GOSUB 270:SE=V
130 INPUT "DESTINATION START ADDRS (HEX)= ":V$
140 GOSUB 270:DS=V
150 INPUT "DEVICE COMMAND LINE          =";V$
160 PRINT #-2,V$
165 FOR I = 1 TO 1000: NEXT I
170 L=SE-SS:IF L>15 THEN L=15'DECIDE LINE LENGTH
180 IF L<0 THEN 260
190 V=L+4:CS=0:PRINT #-2,"S1";GOSUB 390'OPEN LINE
200 V=DS:GOSUB 340'DO ADDRESS
210 FOR SS=SS TO SS+L
220 V=PEEK(SS):GOSUB 390:NEXT SS
230 V=256*INT(CS/256)-CS+255
240 GOSUB 390:PRINT #-2
250 DS=DS+L+1:GOTO 170
260 PRINT #-2,"S9030000FC":END
270 ' HEX STRING TO VALUE SUBROUTINE
280 ' FOR EXTENDED COLOR BASIC,USE INSTEAD
290 ' V=VAL("&H"+V$):RETURN
300 V=0:FOR I=1 TO LEN(V$)
310 V1=ASC(MID$(V$,I,1))-48
320 IF V1>16 THEN V1=V1-7
330 V=V*16+V1:NEXT I:RETURN
340 ' PRINT 4 HEX DIGITS SUBROUTINE
350 V1=INT(V/256):V2=V-256*V1:V=V1
360 GOSUB 390 'DO 2 DIGITS
370 V=V2:GOSUB 390 'DO OTHERS
380 RETURN
390 ' PRINT 2 HEX DIGITS SUBROUTINE
400 CS=CS+V 'DO CHECKSUM
410 ' WITH EXTENDED BASIC, NOW USE
420 ' IF V<16 THEN PRINT #-2,"0";
430 ' PRINT #-2,HEX$(V);:RETURN
440 V1=INT(V/16):V=V-16*V1
450 GOSUB 460:V1=V RETURN
460 PRINT #-2,MID$("0123456789ABCDEF",V1+1,1);:RETURN
```

TRS-80 COLOR COMPUTER/SHOOTER
CONNECTION

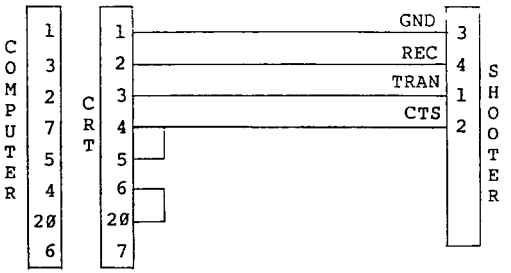


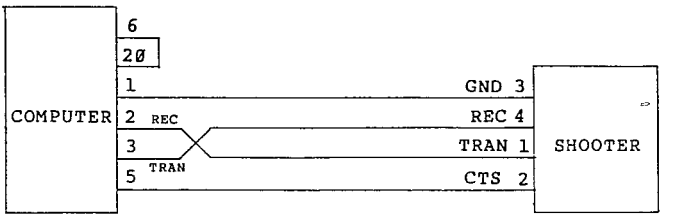
FLEXØ9 INTERFACE SOFTWARE DRIVER

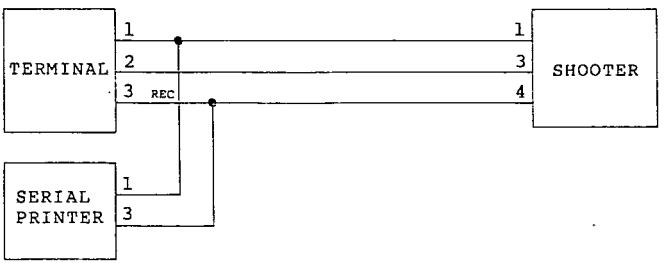
In order to interface the SHOOTER to your SWTPC or equivalant 68Ø9 system you will require a program to handle the serial communications to your I/O port. This type of a program is generally referred to as a "Modem" or "Terminal" program. Since the SHOOTER is terminal oriented, all of the functions can be controlled via a terminal or a computer running under the "Terminal" program. PRO78.BIN program supplied to you is a specialized version of this program which also allows you to transfer a file in the Motorola Hex Format (MIKBUG punch routine or the Kansas City Format). The example of this type of format is provided in the programmer manual.

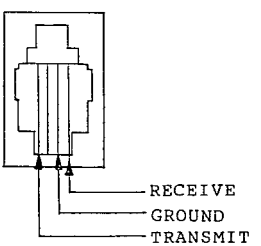
This software driver utilizes the monitor I/O driver routines as well as the punch routine (the routine that takes a section of the memory and formats it to the Motorola Hex File.) Of SWTPC SBUG-E monitor. If you are not using this monitor program you can change the address of the called routines to your version of the monitor. The source listing for the program is provided so that you can customize this software.

Please note that this program must be loaded in memory and executed. To Download to the programmer you must type "T" followed by the starting and ending address of the program memory.

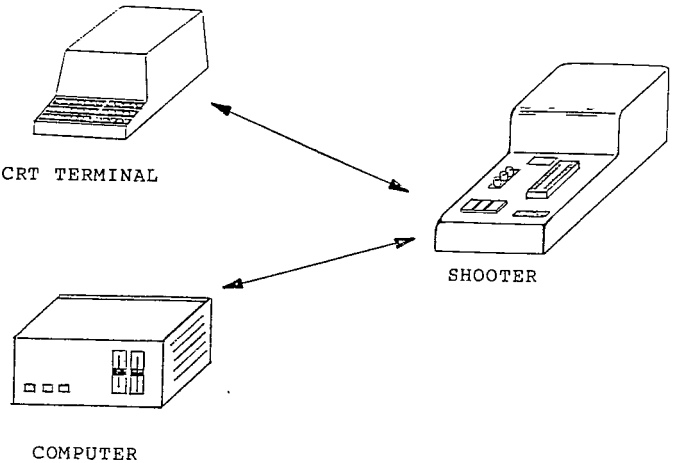




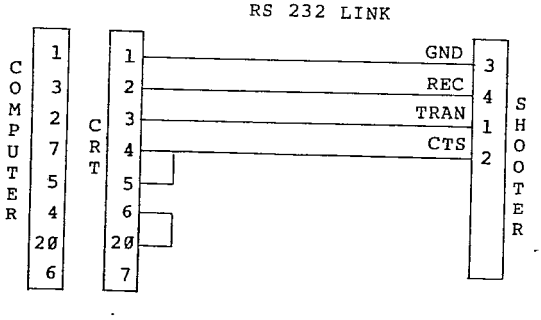




RS-232
SHOOTER
VIEW FROM THE REAR


CRT TERMINAL
SHOOTER
COMPUTER

RS 232 LINK


COMPUTER
1
3
2
7
5
4
20
6
CRT
1
2
3
4
5
6
20
7
GND
REC
TRAN
CTS
3
4
1
2
SHOOTER

THE SERIAL CONNECTION

How to overcome the incompatibilites of your RS232C interface.

The majority of peripheral devices on the market today such as modems, terminals, and printers are offered with a plug compatible method of interconnection known as the RS-232C interface. While this interface is one of the most standard things in the computing industry today, differences between manufacturers and device application cause incompatibilities that must be overcome in actual use. In many cases, devices can be connected with a simple cable, available from Radio Shack or your local computer store, and the devices will function without any problems. In many cases the device will go into paper weight mode and do little more than use electricity. In the second case, the user has two options; hire someone to interface the device to the machine (and at $40 to $75 an hour, this can cost as much as the device), or by a simple procedure, it can be done by the user.

THE RS-232C STANDARD

The IEEE RS-232C standardizes electrical cabling, and data protocol necessary when interconnecting two data devices. The constant electrical voltage and ampreage levels insure that all RS-232C lines can be interconnected without damaging the respective devices. The cabling standard makes these interfaces plug compatible; the data protocol insures that the data sent over the lines will contain the correct number of bits in the correct order at the correct speed.

RS-232C VOLTAGE LEVELS

The RS-232C interface has three types of lines which are signal, handshaking, and ground. The signal lines carry the data between two devices, such as a computer and a terminal, or a terminal and a modem. The ground lines insure an equipment ground, and form a circuit for the signal and handshaking lines. The handshaking lines determine if the devices are ready to transmit and receive data.

SIGNAL: The signal lines are held at approximately -16 VDC. When a data bit comes down the line the voltage is pulled to zero; in some cases it's pulled positive. Therefore, when the signal line is inactive it is negative; when data is being transmitted, the line alternates between negative and ground. For the purpose of interfacing, the most important thing to remember is that the signal lines are negative when compared to line seven, which is signal ground.

GROUND: There are two ground lines available; the signal ground line and the chassis ground line. These are both held at 0 VDC, but line seven is held at zero in respect to the signal and handshaking line. The chassis ground line connects the two devices and protects them from static and voltage biases.

HANDSHAKING: The handshaking lines are positive when ready, and ground or negative indicating a not ready condition. No signals or wave forms are sent over these lines. The simple state of either positive voltage or no voltage is the only activity on these lines. This makes the handshaking lines easy to intermix and differentiate from the signal or ground lines.

Cabling Conventions. The RS-232C cabling convention calls for the use of a standard 25 pin connector referred to as the DB-25. While the DB-25 has 25 lines available, only nine are commonly used; lines one through eight and twenty. The additional pins can be used for interfaces such as the current loop or TTL standards, which are often offered as alternatives on many devices.

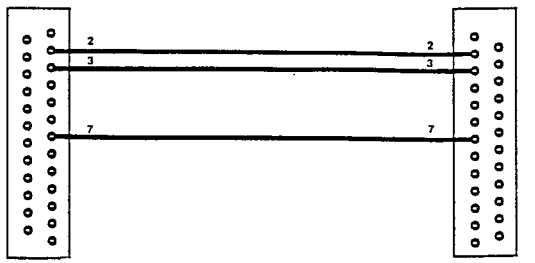
The nine pins can be divided into

the same three groups mentioned above; the signal lines, the handshaking lines and the ground lines.These three groups are distinguished from each other by function, connector position, and electrical characteristics.

Ground. The ground lines are the easiest. When constructing a cable, pin seven is always connected to pin seven, and pin one is always connected to pin one. Pin seven is the signal ground wire and must always be connected. Pin one is the chassis ground. It's the equivalent to the third prong on a houshold electrical plug. Leaving this pin unconnected can cause disaster.

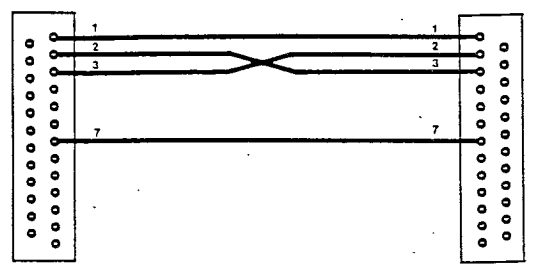
Signal. The signal lines are a bit more difficult. Pin two is transmit data and pin three is receive data. Therefore, the data transmitted by device A should be connected to the receive lines of device B, which means that these lines must often be crossed. There are three methods for determining the placement of the pins. First, consult the documentation, if pin two on both devices is transmit data, lines two and three may have to be crossed. The second method involves trial and error. Since there are only two possible combinations, if everything else is working correctly, alternating the placement should be an easy way to find which pin goes where. The last method involves the use of a tool specifically designed to solve this problem: the breakout box. Plug the breakout box into the circuit and manipulate the switches and jumpers until the indicator LED lights up for both lines two and three.

Handshaking. Aside from the ground lines, the signal lines are often the only requirement for the RS-232C interface. In other words, a cable can be constructed with only lines two, three and seven connected (and frequently is). Many devices have requirements beyond this two wire interface. Modems often require that the device connectd to it show that is connected by bringing DSR (Data Set Ready) active. Active means about +12 volts. After the modem knows that there is something connected to it, it will try to establish communications with another modem over a phone line. When this happens, the modem will bring the CD (Character Detect) line active and the device will know that is is OK to transmit and receive data over the modem link.



DEVICE A		DEVICE B	
PIN	FUNCTION	PIN	FUNCTION
2	Send data	2	Receive data
3	Receive data	3	Send data
7	Ground	7	Ground

Figure A
The RS-232C three wire interface is the minimum interface with which a device will function. It contains receive data, transmit data, and signal ground.



DEVICE A		DEVICE B	
PIN	FUNCTION	PIN	FUNCTION
1	Frame Ground	1	Frame Ground
2	Send data	2	Receive data
3	Receive data	3	Send data
7	Ground	7	Ground

Figure B
The three wire cross-over cable is a frequent modification needed to hook up devices with non-compatible pin connections.

As a more specific example, let's say we have a computer connected to a modem and that line two is transmit data on the computer and line two is receive data on the modem, and they are connected. By the same token, line three is transmit data on the modem and line three is receive data on the computer. When the wall power is turned on to both devices, the computer causes line twenty (DSR) to become active. DSR on the modem end is line six. When we connect line twenty on the computer to line six on the modem, the modem knows that the computer is connected.

When the modem sets up a carrier detect link, it raises line eight to an active state. Line eight on the modem is connected to line eight on the computer and lets the computer know that the phone connection is valid. A valid modem-computer link is now made.

Another case might involve a modem and a terminal. In this case, both the terminal and the modem use line two as transmit data, and both use line three as receive data. These lines

eight representing parity. Here again, the amount of data bits between the two devices should match.

HOW TO TROUBLE SHOOT YOUR INTERFACE PROBLEM

The RS-232C is an indestructible interface. Short of plugging it into a wall socket, the equipment probably won't be damaged by experimenting. If it doesn't work, it can be played with until it does work. Here are some helpful hints.

Problem: The power is on, the devices are connected, but the printer won't print.

Solution: There is probably a cabling problem. Switch lines two and three. If that doesn't work, switch lines four and five. If you're still not successful, short lines six, eight and twenty together.

Problem: The printer prints out the correct number of characters, but half or all of them are not what they are supposed to be.

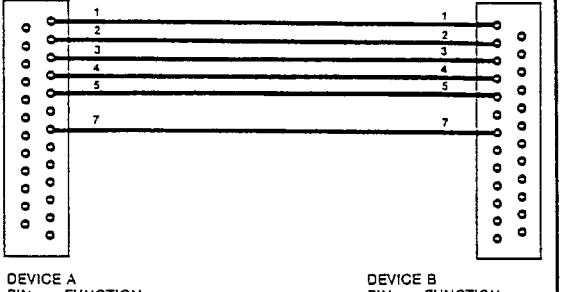
Solution: The parity bits aren't properly set.

Problem: The printer prints out garbage and the number of garbage characters doesn't match the number of characters sent to the printer.

Solution: The baud rate is incorrect.

CONCLUSION

A common quality among data processors is persistence. Most software people will work on a sofware bug for a week and never complain. Well, maybe once. . . . That same type of persistence is often needed to overcome RS-232C problems. They won't always work the first time, but they will always work.



DEVICE A		DEVICE B	
PIN	FUNCTION	PIN	FUNCTION
1	Frame Ground	1	Frame Ground
2	Send data	2	Receive data
3	Receive data	3	Send data
4	Request to send	4	Request to send
5	Clear to send	5	Clear to send
7	Ground	7	Ground

Figure E

The 202 protocol modem can only transmit data in one direction so the device it is connected to must bring line four active when it wants to transmit and wait for line five to become active.

GENERAL EPROM/EPROM PROGRAMMING HANDBOOK

WHAT ARE EPROMs ?

THIS QUESTION MAY BE A VERY OBVIOUS TO ALMOST ANYONE WHO HAS PURCHASED A PROGRAMMER. HOWEVER FOR THE VERY RECENT TO THE FIELD, EPROMs ARE MOS SEMICONDUCTOR MEMORIES USED MOSTLY IN CONJUNCTION WITH A MICROPROCESSOR. THE INSTRUCTIONS AND DATA FOR THE MICRO (Z-80 ,8080, 6800 AS EXAMPLES) ARE STORED PERMANENTLY IN THE EPROM. EPROM IS THE ACRONYM FOR ERASABLE PROGRAMMABLE READ ONLY MEMORY. ONCE A UV EPROM HAS BEEN PROGRAMMED IT CAN ONLY BE ERASED BY AN ULTRAVIOLET SOURCE(LOGICAL DEVICES QUV-T8 MODEL OR EQUIVALENT). FOR THIS REASON MOST COMPUTERS USE EPROMS TO STORE VITAL COMPUTER INSTRUCTIONS THAT CANNOT BE LOST AFTER POWER-DOWN.
SINCE EPROMS CANNOT BE WRITTEN TO BY THE MICROPROCESSOR DIRECTLY THE COMPUTER SYSTEM WILL NEED RAM AND DISK MEMORY FOR TEMPORARY STORAGE AS WELL AS APPLICATION DATA AND PROGRAMS.
MANY COMPANIES USE EPROMS TO STORE APPLICATION PROGRAMS SUCH AS GAMES OR COMPUTER UTILITY PROGRAMS ON EPROMS PACKAGED IN CARTRIDGES. THESE EPROMS CONTAIN MACHINE LANGUAGE INSTRUCTIONS FOR THE PARTICULAR MICROPROCESSOR USED IN THE COMPUTER. USER OFTEN NEED NOT TO BE CONCERNED WITH THE CONTENTS OF THESE EPROMS. HOWEVER FOR THE TECHNICALY INCLINED WHO WANTS TO DECIPHER THE CODE AND MAKE HIS VERSION OF THE PROGRAM THE EPROM MUST BE READ INTO AN EPROM PROGRAMMER (SUCH AS THE LOGICAL DEVICES PROMPRO SERIES). ONCE THE EPROM HAS BEEN READ INTO THE PROGRAMMER, THE DATA CAN BE SENT-UP TO A COMPUTER FROM THE SERIAL PORT FOR STORAGE ON THE DISK DRIVE.
AT THIS POINT USER MUST BE VERY FAMILIAR WITH THE LANGUAGE OF THE MICROPROCESSOR. THE UPLOADED FILE CAN BE DISASSEMBLED BY A SOFTWARE DEVELOPMENT TOOL CALLED A "DISASSEMBLER". THIS PROCESS CONVERTS THE HEX MACHINE INSTRUCTIONS TO A SERIES OF TEXT INSTRUCTION MNEMONICS REPRESENTING THE PROGRAM CODE. THIS ASSEMBLED CODE CAN BE ENTERED IN A TEXT EDITOR OR A WORD PROCESSOR FOR ALTERATIONS. ONCE THE CHANGES HAVE BEEN MADE TO THE CODE, THE PROGRAM CAN BE CONVERTED BACK TO MACHINE CODE LEVEL BY USING A SOFTWARE DEVELOPMENT TOOL CALLED AN "ASSEMBLER". THE OUTPUT OF THE ASSEMBLER THAT IS GENERALLY EITHER IN AN ABSOLUTE HEX OR IN A FORMATTED HEX FORM IS STORED BACK ON THE DISK. THIS FILE NOW IS TO BE SENT BACK TO THE PROGRAMMER BY AN OPERATION CALLED "DOWNLOAD". THE EPROM PROGRAMMER GENERALLY ACCEPTS A FILE IN A FORMATTED HEX FORM. THERE ARE TWO INDUSTRY STANDARDS; INTEL HEX OR MOTOROLA HEX. BOTH FORMATS ARE SIMILAR.(REFER TO SECTION XXX FOR MORE DETAILED DESCRIPTION). AFTER THE PROMPRO PROGRAMMER RECEIVES THE ASCII HEX FILE IT AUTOMATICALLY EXTRACTS THE DATA SECTION(MACHINE HEX CODE) AND STORES IT IN ITS RAM BUFFER. A BLANK EPROM (ONE THAT CONTAINS ALL "1" PATTERN") IS PLACED ON THE PROGRAMMER SOCKET AND THE RAM DATA IS PROGRAMMED ON THE EPROM. AFTER SUCCESSFUL PROGRAM OPERATION

THE EPROM IS REMOVED AND PLACED EITHER IN THE CARTRIDGE OR THE EPROM SOCKET INSIDE OF THE COMPUTER. HOPEFULLY THAT YOU DID NOT MAKE ANY MISTAKES IN WRITTING YOUR PROGRAM, AND AFTER YOU HAVE POWERED UP YOUR COMPUTER EVERYTHING WORKS TO YOUR SATISFACTION.
YOU MUST MAKE SURE THAT ANYTIME THAT YOU DEAL WITH ANOTHER COMPANIES SOFTWARE PROGRAM THAT YOU UNDERSTAND ALL OF THE COPYRIGHT LAWS AND BY ALTERING THOSE PROGRAMS YOU ARE NOT VIOLATING ANY OF THOSE LAWS.
EPROMS ARE FABRICATED IN A VARIETY OF MEMORY SIZES HOWEVER OFTEN IN AN 8-BIT ORGANIZATION. THE STANDARD MEMORY SIZES AVAILABLE TODAY ARE THE FOLLOWING;

GENERIC TYPE	MEMORY SIZE	PACKAGE
2716	2048 x 8	24 PIN
2732	4096 x 8	24 PIN
2764	8192 x 8	28 PIN
27128	16384 x 8	28 PIN
27256	32768 x 8	28 PIN

EPROMS ALSO ARE AVAILABLE WITH SEVERAL CHOICES OF ACCESS-TIME SPEEDS. THE DASH NUMBER AFTER THE GENERIC TYPE DESIGNATION ON THE EPROM OFTEN INDICATES THE SPEED. YOU MUST CONSULT THE DATA SHEET FOR THE PARTICULAR MANUFACTURER TO INSURE THAT THE EPROM YOU ARE USING MEETS THE ACCESS-TIME REQUIRED BY YOUR COMPUTER.
THE PROMPRO EPROM PROGRAMMING EQUIPMENT IS INDIFFERENT TO THE EPROM ACCESS-TIME AND CAN PROGRAM ANY SPEED EPROM.

EPROMS CAN BE ERASED AND REPROGRAMMED MANY TIMES, ALTHOUGH AS THE NUMBER OF ERASER TIME INCREASES SO DOES THE TIME REQUIRED TO ERASE THEM. A NEW EPROM WITH A GOOD QUALITY QUARTZ WINDOW CAN TAKE AS LITTLE AS 7 MINUTES TO ERASE WITH A QUV-T8/Z EPROM ERASER.

MOST OF THE EPROMS TODAY OPERATE WITH A POWER SUPPLY VOLTAGE (VCC) OF 4.75 - 5.25 WITH 5.00 NOMINAL VALUE. VCC IS APPLIED TO PIN 24 FOR 24 PIN PACKAGES AND PIN 28 FOR 28 PIN PACKAGES.
IN ORDER TO PROGRAM AN EPROM IT IS REQUIRED TO APPLY A HIGH VOLTAGE TO ONE OF THE SIGNAL PINS(PROG) AND PULSE ONE OTHER OR THE SAME PIN FOR A PERIOD OF 50 MILLISECONDS, WHILE APPLYING THE PROPER ADDRESS AND DATA TO THE EPROM.

EPROMS ARE EXTREMELY SENSITIVE TO STATIC ELECTRICITY. IT IS IMPERATIVE THAT YOU ALWAYS USE CONDUCTIVE FOAM PAD WHEN TRANSPORTING EPROMS (LOGICAL PN# AP-10). YOU MUST ALSO USE A GROUNDING STRAP OR GROUNDING BAR WHENEVER WORKING DIRECTLY WITH EPROMS OR THE PROGRAMMING EQUIPMENT.

MOST EPROMS WHEN SHIPPED FROM THE FACTORY ARE ALREADY ERASED (ALL "FF PATTERN). IT IS A GOOD IDEA HOWEVER TO CHECK EPROMS PRIOR TO PROGRAMMING FOR THE BLANK PATTERN.

EPROMS ARE ALSO VERY SENSITIVE TO POWER GLITHES. IF YOU ARE HAVE A SEVERE STORM IN THE AREA OR UNUSUALLY HIGH POWER INTERRUPTIONS IT IS BEST TO POSTPONE PROM PORGRAMMING UNTIL THE PROBLEM IS CLEARED. REMOVE YOUR EPROMS FROM THE EQUIPMENT DURING POWER-UP AND POWER-DOWN.

FADING. FADING OCCURS ON EPROMS THAT ARE POORLY PROGRAMMED OR ARE DEFECTIVE. EPROMS REQUIRE A CERTAIN MINIMUM NUMBER OF ELECTRICAL CHARGES IN THE FLOATING GATE OF THE FET SWITCHES. THESE CHARGES TEND TO LEAK OUT GRADUALLY OVER A LONG PERIOD OF TIME. IF THE EPROM PROGRAMMING EQUIPMENT DOES NOT SUPPLY THE REQUIRED AMOUNT OF CHARGES, THE EPROM MAY INITIALY PROGRAM AND VERIFY PROPERLY AND WORK IN YOUR COMPUTER FOR SEVERAL MONTHS, UNTIL THE EPROM BITS START CHANGING STATE INTERMITTENTLY. SOME MANUFACTURERS OF EPROM PROGRAMMING EQUIPMENT MAY PROMISE FAST PROGRAM TIME ON EPROMS TO SELL THEIR EQUIPMENT, WITHOUT CONSIDERATIONS TO THE PROPER PROGRAMMING ALGORITHMS. LOGICAL DEVICES INC. ADHERES VERY STRONGLY TO THE MANUFACTURERS SPECIFICATIONS TO MAXIMIZE THE FIELD RELIABILITY AND LONGEVITY OF YOUR EPROMS.

EPROM TYPES. WITHIN THE SAME GENERIC FAMILY THERE MAY BE SEVERAL TYPE OF EPROMS THAT VARY AS FAR AS THE PROGRAMMING VOLTAGE REQUIREMENTS. THERE ARE PRESENTLY TWO COMMON PROGRAMMING VOLTAGE LEVELS; 25 VOLTS AND 21 VOLTS. SOME NEW DEVICES WILL REQUIRE 12.5 PROGRAMMING VOLTAGE.
FOR EXAMPLE A 2732 EPROMS REQUIRES A 25 VOLT PROGRAM VOLTAGE WHEREAS, A 2732A REQUIRES 21 VOLTS TO PROGRAM. ATTEMPTING TO PROGRAM A 2732A WITH 25 VOLT CAN DESTROY THE DEVICE.
EPROMS HAVE A MUCH BETTER PROGRAMMING YIELD THAN BIPOLAR PROMS. USER MUST HAVE BETTER THAT 5% PROGRAMMING YIELD FOR EPROMS. IF THE FAILURE RATES EXCCEED 5% USER MUST SUSPECT THE EPROMS OR THE EQUIPMENT.
WHERE CAN YOU BUY EPROMS?. IT IS RECOMMENDED THAT YOU SELECT RELIABLE SOURCES FOR YOUR COMPONENTS. AVOID BUYING EPROMS FROM UNCERTAIN ORIGINS OR FROM A QUESTIONABLE SURPLUS STORE. LOGICAL DEVICES INC. CAN RECOMMEND THE FOLLOWING SOURCES FOR YOUR EPROMS;

CASPIAN TECHNOLOGY
781 W. OAKLAND PRK BLVD.
SUITE 580
FT. LAUDERDALE FL 33311
TEL; 305-974-0977

HAMILTON AVNET
HALLMARK
ARROW ELECTRONICS
HAMMOND ELECTRONICS

PLEASE CHECK YOUR LOCAL TELEPHONE BOOK FOR LOCAL BRANCH OFFICE OF THE ABOVE DISTRIBUTORS.

MANUFACTURERS OF PROGRAMMABLE READ-ONLY MEMORIES

ADVANCED MICRO DEVICES
901 THOMPSON PL
SUNNYVALE, CA 94086
(408) 732-2400

AMERICAN MICROSYSTEMS INC
3800 HOMESTEAD RD
SANTA CLARA, CA 95051
(408) 246-0330

ELECTRONIC ARRAYS
550 E MIDDLEFIELD R
MT VIEW, CA 94043
(415) 964-4321

FAIRCHILD SEMICONDUCTOR
464 ELLIS ST
MT VIEW, CA 94042
(415) 962-5011

FUJITSU MICROELECTRONICS
2945 OAKMEAD VILLAGE CT
SANTA CLARA, CA 95051
(408) 729-1700

GEN INSTRUMENT CORP
MICROELECTRONIC DIV
600 W JOHNS ST
HICKSVILLE, NY 11802
(516) 733-3000

HARRIS SEMICONDUCTOR
BOX 883
MELBOURNE, FL 32901
(305) 724-7407

HITACHI AMERICA LTD
1800 BERING DR
SAN JOSE, CA 95112
(408) 292-6404

HUGHES AIRCRAFT CO
SOLID ST PROD DIV
500 SUPERIOR AVE
NEWPORT BCH, CA 926
(714) 759-2411

INTEL CORP
3065 BOWERS AVE
SANTA CLARA, CA 95051

INTERSIL INC
10710 N TANTAU AVE
CUPERTINO, CA 95014

MOSTEK CORP
1215 W CROSBY DR
CARROLLTON, TX 75006

MITSUBISHI/MEICO
3030 E VICTORIA ST
COMPTON, CA 90221
(213) 537-7131

MONOLITHIC MEMORIES INC
1165 E ARQUES AVE
SUNNYVALE, CA 94086
(408) 739-3535

MOTOROLA INC
INTERGRATED CIRC DIV
3501 ED BLUESTEIN BLV
AUSTIN, TX 78721
(512) 928-6000

NATIONAL SEMICONDUCTOR
2900 SEMICONDUCTOR DR
SANTA CLARA, CA 95051
(408) 737-5000

NEC MICROCOMPUTERS INC
173 WORCESTER ST
WELLESLEY, MA 02181
(617) 239-1910

NITRON INC
10420 BUBB RD
CUPERTINO, CA 95014
(408) 255-7550

OKI SEMICONDUCTOR
SUITE 405
SANTA CLARA, CA 95051
(408) 984-4842

PLESSEY SEMICONDUCTOR
1641 KAISER AVE
IRVINE, CA 92714
(714) 540-9979

PANASONIC
1 PANASONIC WAY
SECAUCUS, NJ 07094
(201) 348-7276

RAYTHEON SEMICONDUCTOR
350 ELLIS ST
MT VIEW, CA 94042
(415) 968-9211

SGS-ATES SEMICONDUCTOR CRP
240 BEAR HILL RD
WALTHAM, MA 02154
(617) 890-6688

RCA SOLID STATE DIV
RTE 202
SOMERVILLE, NJ 08876
(201) 685-6000

SIGNETICS CORP
811 E ARQUES AVE
SUNNYVALE, CA 94086
(408) 739-7000

TEXAS INSTRUMENTS INC
BOX 225012, M/S 308
DALLAS, TX 75265

SYNERTEK
BOX 552
SANTA CLARA, CA 95051
(408) 988-5611

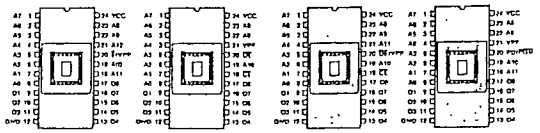
TOSHIBA AMERICA INC
2151 MICHELSON DR
IRVINE, CA 92715
(714) 955-1155

XICOR INC
1221 INNSBRUCK DR
SUNNYVALE, CA 94086
(408) 734-3041

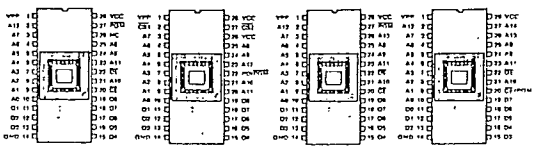
ZILOG
10460 BUBB RD
SUNNYVALE, CA 94086
(408) 446-4666

EPROM PIN-OUTS

MCM 6876A, 68766 2516, 2716, 27C16 2732, 27C32, 2732A 25L32, 2532



27BA, 27C64, 2764A 2564 27128, 27128A 27256



EPROM PIN-OUT GUIDE

EPROM TYPE	VCC PIN	GRND PIN	VPP PIN	VPP VOLTAGE	PGM PULSE	PGM WITH PULSE	PIN OUTPUT ENABLE
2716	24/26	12/14	21/23	25V ± 1V	18/20	50ms	20/22
2732	24/26	12/14	20/23	25V ± 1V	20/22	50ms	20/22
2732A	24/26	12/14	20/22	21V ± 1V	20/22	50ms	20/22
2764	28	14	1	21V ± v	27	50ms	20/22
2564	28	14	1	21V ± V	22	50ms	20/22
27128	28	14	1	21V ± V	27	50ms	20/22
27256	28	14	1	12.5V ± 5V	27	*	20/22
6876A	24/26	12/14	20/22	25V ± 1V	20/22	*	20/22
2532	24/26	12/14	20/22	25 ± 1V	20/25	50ms	20/22

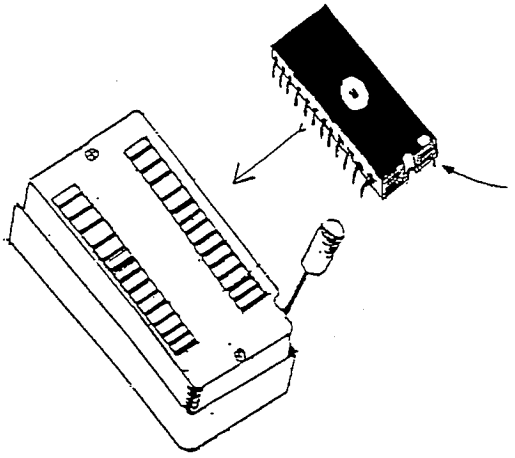
PIN NUMBERS ARE BOTH INDICATED FOR BOTH 24 PIN AND 28 PIN SOCKETS BY THE "/".

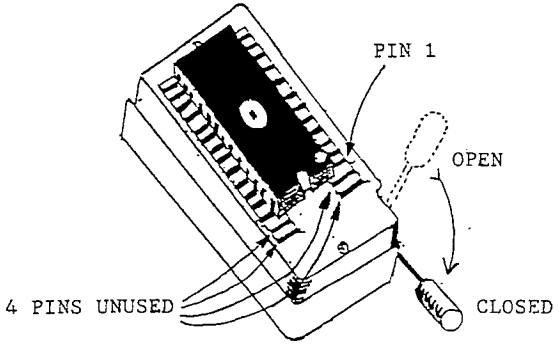
2764A, 27128A 12.5 VPP
VPP = PROGRAM VOLTAGE
VCC = SUPPLY CURRENT 5V ± .25V

LEADER IN PROGRAMMING EQUIPMENT:

LOGICAL DEVICES INC.

INSERTING A 24 PIN DEVICE IN SOCKET 1





ZIF SOCKET 1

DEAR CUSTOMER

Please make any comments that you may have about the manual in the space provided below and return to us as soon as possible. Your comments will be reviewed and considered for the newer version manuals.

Please be specific in regards to your comments and specify areas of the manual that are either hard to read, unclear or poorly organized. Your feedback is greatly appreciated.

Please be sure to fill in your name and address below so that we can send you any future updates.

COMMENTS

Name:
Company:
Address:

Telephone: Date: